SIM65M Series_NMEA Message_User Guide

GNSS Module

SIMCom Wireless Solutions Limited
SIMCom Headquarters Building, Building 3, No. 289 Linhong Road, Changning District, Shanghai P.R. China
Tel: 86-21-31575100
support@simcom.com
www.simcom.com

Document Title:	SIM65M Series_NMEA Message_User Guide
Version:	1.00
Date:	2022.07.21
Status:	Release

GENERAL NOTES

SIMCOM OFFERS THIS INFORMATION AS A SERVICE TO ITS CUSTOMERS, TO SUPPORT APPLICATION AND ENGINEERING EFFORTS THAT USE THE PRODUCTS DESIGNED BY SIMCOM. THE INFORMATION PROVIDED IS BASED UPON REQUIREMENTS SPECIFICALLY PROVIDED TO SIMCOM BY THE CUSTOMERS. SIMCOM HAS NOT UNDERTAKEN ANY INDEPENDENT SEARCH FOR ADDITIONAL RELEVANT INFORMATION, INCLUDING ANY INFORMATION THAT MAY BE IN THE CUSTOMER'S POSSESSION. FURTHERMORE, SYSTEM VALIDATION OF THIS PRODUCT DESIGNED BY SIMCOM WITHIN A LARGER ELECTRONIC SYSTEM REMAINS THE RESPONSIBILITY OF THE CUSTOMER OR THE CUSTOMER'S SYSTEM INTEGRATOR. ALL SPECIFICATIONS SUPPLIED HEREIN ARE SUBJECT TO CHANGE.

COPYRIGHT

SIMCom Wireless Solutions Limited
SIMCom Headquarters Building, Building 3, No. 289 Linhong Road, Changning District, Shanghai P.R. China
Tel: +86 21 31575100
Email: simcom@simcom.com

For more information, please visit:
https://www.simcom.com/download/list-863-en.html

For technical support, or to report documentation errors, please visit:
https://www.simcom.com/ask/ or email to: support@simcom.com

About Document

Version History

Version	Date	Owner	What is new
V1.00	2020.07.21	Jianying.hu/Wenjie.lai	Original

Scope

This document applies to the following products

Name	Type	Size(mm)	Comments
SIM65M			N/A

Contents

1 Introduction

1.1 Purpose of the document

At present, has been built and is planning the construction of a satellite navigation system apart from United States GPS system, and Russia's GLONASS system, the European Galileo system, Beidou satellite navigation system in China and Japan and Indian regional satellite navigation systems.

This document will introduce GNSS NMEA Message application process.

Developers could understand and develop application quickly and efficiently based on this document.

1.2 Related documents

1.3 Conventions and abbreviations

2 NMEA Messages

2.1 General Format of NMEA Messages

NMEA messages use the ASCII character set and have a defined format. Each message begins with a $ (hex 0x24) and end with a carriage return and line feed (hex 0x0D 0x0A, represented as <CR><LF>). Each message consists of one or more fields of ASCII letters and numbers, separated by commas. After the last field, and before the <CR><LF> is a checksum consisting of an asterisk (*, hex 0x2A) followed by two ASCII characters representing the hexadecimal value of the checksum. The checksum is computed as the exclusive OR of all characters between the $ and * characters.

Parameter	Example	Contents
Preamble	$	
TalkerID	GP	It is used for various GNSS configurations, such as GP/GL/GA/GB/GN.
SentenceID	RMC	Fields descriptions, such as GGA/GSA/GSV/RMC.
Payload	<Data>	Message specific data. Refer to a specific message section for <data>...<data> definition
Checksum	*CKSUM	CKSUM is a two-hex ASCII character. Checksums is required in all input messages
End	<CR> <LF>	Each message is terminated using Carriage Return (CR) Line Feed (LF) which are \r\n. Because \r\n are not printable ASCII characters, they are omitted from the example strings, but must be sent to terminate the message and cause the receiver to process that input message

Talker ID description.

Talker ID	Description (Configuration GNSS)
GP	GPS
GL	GLONASS
GA	Galileo
GB*	Beidou
GI*	NavIC
GN	Multi-GNSS

* NMEA v3.01/v4.10 does not define talker ID for Beidou/NavIC. ‘GB’/’GI’ only defines in NMEA v4.11.

Sentence ID description

Sentence ID	Description
GGA	Global Positioning System Fix Data
GLL	Geographic Position, Latitude and Longitude
GSA	GNSS DOP and Active Satellites
GSV	GNSS Satellites In View
RMC	Recommended Minimum Specific GNSS Data
VTG	Course Over Ground & Ground Speed
ZDA	GNSS Time & Date

Talker ID display in different GNSS system (for NMEA 0183 v3.01).

Talker ID	GPS only	GLONASS only	Galileo only	Beidou only	NavIC only	Multi-GNSS GPS+GLO+GAL+BDS+NavIC
GGA	GP	GL	GA	GB*	GI*	GN
GLL						
RMC						
VTG						
ZDA						
GSA**						GP+GL+GA+GB+GI
GSV						

Talker ID display in different GNSS system (for NMEA 0183 v4.10).

Talker ID	GPS only	GLONASS only	Galileo only	Beidou only	NavIC only	Multi-GNSS GPS+GLO+GAL+BDS+NavIC
GGA	GP	GL	GA	GB*	GI*	GN
GLL						
RMC						
VTG						
ZDA						
GSA**						
GSV						GP+GL+GA+GB+GI

* NMEA v3.01/v4.10 does not define talker ID for Beidou/NavIC. ‘GB’/’GI’ only defines in NMEA v4.11.

** The difference between NMEA 0183 v3.02 and v4.10 for talker ID is GSA.

System/Signal ID in NMEA sentence

Constellation	System ID (AIROHA)	Signal ID (AIROHA)	System ID (NMEA 0183 v4.10)	Signal ID (NMEA 0183 v4.10)
GPS L1C/A	1	1	1	1
GPS L5Q	1	8	1	8
GLONASS L1	2	1	2	1
Galileo E1-BC	3	7	3	7
Galileo E5a	3	1	3	1
Beidou B1I	4*	1*		
Beidou B2a	4*	4*		

NavIC L5	6*	1*		

* Beidou/NavIC is not defined in NMEA v4.10

Satellite ID in NMEA sentence

Constellation	PRN numbers	Satellite ID (AIROHA)	Satellite ID (NMEA 0183 v4.10)
GPS	1-32	1-32	1-32
SBAS	120-138	33-51	33-64
GLONASS	1-24	65-88	65-99
Galileo	1-36	1-36	1-36
Beidou	1-63	1-63	N/A
QZSS	193-199	193-199	N/A
NavIC	1-14	1-14	N/A

NOTE

- All fields in all proprietary NMEA messages are required, none are optional and are comma delimited
- In some numeric fields representing a single data element, leading zeros before a decimal are suppressed. A single "0" character preceding the decimal point is maintained. In compound numeric structures (such as LAT or LONG), leading zeros are suppressed only on the leftmost element Trailing zeros are not suppressed

2.2 Standard NMEA Output Messages

The standard NMEA messages are GGA/GLL/GSA/GSV/RMC/VTG/ZDA. The satellite ID, system ID, and signal ID in the NMEA sentences are defined in section 2.1. The following shows the details of these messages based on NMEA 0183 v4.10.

A full description of the listed NMEA messages is provided in the following sections.

2.2.1 Message ID GGA: Global Positioning System Fixed Data

Example:
$GPGGA,091926.000,3113.3166,N,12121.2682,E,1,09,0.9,36.9,M,7.9,M,,0000*56<CR><LF>

Name	Example	Unit	Description
Message ID	$GPGGA		GGA protocol header
UTC Time	091926.000		hhmmss.sss
Latitude	3113.3166		ddmm.mmmm
N/S Indicator	N		N=north or S=south
Longitude	12121.2682		dddmm.mmmm
E/W Indicator	E		E=east or W=west
Position Fix Indicator	1		See Table 2.2.1
Satellites Used	09		Range 0 to 12
HDOP	0.9		Horizontal Dilution of Precision
MSL Altitude	36.9	meters	
Units	M	meters	
Geoid Separation	7.9	meters	Geoid-to-ellipsoid separation. Ellipsoid altitude = MSL Altitude + Geoid Separation.
Units	M	meters	
Age of Differential Correction		sec	Null fields when DGPS is not used
Differential Refference Station ID	0000		
Checksum	*56		
<CR><LF>			End of message termination

Table 2.2.1

Value	Description
0	Fix not available or invalid
1	GPS SPS Mode, fix valid
2	Differential GPS, SPS Mode, fix valid

3	Not supported
4	RTK fixed
5	RTK float
6	Dead Reckoning Mode, fix valid

NOTE

- A valid status is derived from all the parameters set in the software. This includes the minimum number of satellites required, any DOP mask setting, presence of DGPS corrections, etc. If the default or current software setting requires that a factor is met, then if that factor is not met, the solution will be marked as invalid
- We will adjust the number of satellites participating in positioning according to the quality of relevant measurement. Not all L1+L5 satellites tracked will participate in positioning.
  You cannot calculate a very accurate number of users according to NMEA sentence. It is recommended to output the number directly with GGA.

2.2.2 Message ID GLL: Geographic Position - Latitude/Longitude

Example:
$GPGLL,3113.3157,N,12121.2684,E,094051.000,A,A*59<CR><LF>

Name	Example	Unit	Description
Message ID	$GPGLL		GLL protocol header
Latitude	3113.3157		ddmm.mmmm
N/S Indicator	N		N=north or S=south
Longitude	12121.2684		dddmm.mmmm
E/W Indicator	E		E=east or W=west
UTC Time	094051.000		hhmmss.sss
Status	A		A=data valid or V=data not valid
Mode	A		A=Autonomous D=DGPS
Checksum	*59		
<CR><LF>			End of message termination

NOTE

- Position was calculated based on one or more of the SVs having their states derived from almanac parameters, as opposed to ephemerides

2.2.3 Message ID GSA: GNSS DOP and Active Satellites

Example:
$GPGSA,A,3,07,02,26,27,09,04,15, , , , , ,1.8,1.0,1.5*33<CR><LF>
$GNGSA,A,3,13,08,21,15,26,07,01,,,,,,1.19,0.61,1.02,3*01<CR><LF>

Name	Example	Unit	Description
Message ID	$GPGSA		GGA protocol header
Mode 1	A		See Table 2.2.3
Mode 2	3		See Table 2.2.4
Satellite Used [1]	07		SV on Channel 1
Satellite Used [1]	02		SV on Channel 2
....			....
Satellite Used [1]			SV on Channel 12
PDOP[2]	1.8		Position Dilution of Precision
HDOP[2]	1.0		Horizontal Dilution of Precision
VDOP[2]	1.5	meters	Vertical Dilution of Precision
GNSS System ID	1		GNSS System ID(Only supported in NMEA v4.10 format) See Section 2.1
Checksum	*33		
<CR><LF>			End of message termination

NOTE

- Satellite used in solution
- Maximum DOP value reported is 50. When value 50 is reported, the actual DOP may be much larger

Table 2.2.3

Value	Description
M	Manual – Forced to operate in 2D or 3D mode
A	2D Automatic – Allowed to automatically switch 2D/3D

Table 2.2.4

Value	Description
1	Fix not available

2	2D (<4 SVs used)
3	3D (>3 SVs used)

Table 2.2.5 GNSS System ID

Value	Description
1	GPS
2	GLONASS
3	Galileo
4	Beidou
6	NavIC

(Only supported in NMEA v4.10 format, See Section 2.1)

System/Signal ID in NMEA sentence

Constellation	System ID (AIROHA)	Signal ID (AIROHA)	System ID (NMEA 0183 v4.10)	Signal ID (NMEA 0183 v4.10)
GPS L1C/A	1	1	1	1
GPS L5Q	1	8	1	8
GLONASS L1	2	1	2	1
Galileo E1-BC	3	7	3	7
Galileo E5a	3	1	3	1
Beidou B1I	4*	1*		
Beidou B2a	4*	4*		
NavIC L5	6*	1*		

2.2.4 Message ID GSV: GNSS Satellites in View

Example:

$GPGSV,3,1,11,26,68,023,37,15,64,251,33,05,45,058,34,29,33,253,33*75<CR><LF>

$GPGSV,3,2,11,27,32,164,30,21,25,315,29,02,24,140,31,08,19,048,29*70<CR><LF>

$GPGSV,3,3,11,09,16,180,25,18,08,284,27,10,08,085,18*4E<CR><LF>

$GPGSV,4,1,13,20,65,239,41,195,56,093,41,02,54,334,42,193,51,105,41,1*61<CR><LF>

$GAGSV,3,1,09,13,56,005,40,08,44,106,41,21,38,252,40,07,27,168,36,7*73<CR><LF>

Name	Example	Unit	Description

Message ID	$GPGSV		GSV protocol header
Number of Messages [1]	2		Total number of GSV messages to be sent in this group
Message Number[1]	1		Message number in this group of GSV messages
Satellites in View[1]	11		
Satellite ID	26		Channel 1 (Range 1 to 32)
Elevation	68	degrees	Channel 1 (Maximum 90)
Azimuth	023	degrees	Channel 1 (True, Range 0 to 359)
SNR (C/N0)	37	dBHz	Range 0 to 99, null when not tracking
....			....
Satellite ID	29		Channel 4 (Range 1 to 32)
Elevation	33	degrees	Channel 4 (Maximum 90)
Azimuth	253	degrees	Channel 4 (True, Range 0 to 359)
SNR (C/N0)	33	dBHz	Range 0 to 99, null when not tracking
Signed ID	1		Signal ID (Only support in NMEA v4.10 format, See Section 2.1)
Checksum	*75		
<CR><LF>			End of message termination

NOTE

- Depending on the number of satellites tracked, multiple messages of GSV data may be required In some software versions, the maximum number of satellites reported as visible is limited to 12, even though more may be visible

2.2.5 Message ID RMC: Recommended Minimum Specific GNSS Data

Example:
$GPRMC,094330.000,A,3113.3156,N,12121.2686,E,0.51,193.93,171210,,,A*68<CR><LF>
$GNRMC,034212.000,A,2929.4571,N,10638.0646,E,0.00,211.84,130821,,,A,V*04<CR><LF>

Name	Example	Unit	Description
Message ID	$GPRMC		RMC protocol header
UTC Time	094330.000		hhmmss.sss
Status [1]	A		A=data valid or V=data not valid
Latitude	3113.3156		ddmm.mmmm

N/S Indicator	N		N=north or S=south
Longitude	12121.2686		dddmm.mmmm
E/W Indicator	E		E=east or W=west
Speed Over Ground	0.51	knots	
Course Over Ground	193.93	degrees	True
Date	171210		ddmmyy
Magnetic Variation [2]		degrees	E=east or W=west
East/West Indicator[2]			E=east
Mode	A		A=Autonomous D=DGPS
Navigational Status	A		Navigational Status (Only support in NMEA v4.10 format
Checksum	*68		
<CR><LF>			End of message termination

NOTE

- A valid status is derived from all the parameters set in the software. This includes the minimum number of satellites required, any DOP mask setting, presence of DGPS corrections, etc. If the default or current software setting requires that a factor is met, then if that factor is not met, the solution will be marked as invalid
- Does not support magnetic declination. All “course over ground” data are geodetic WGS84 directions relative to true North

2.2.6 Message ID VTG: GNSS DOP and Active Satellites

Example:
$GPVTG,83.37,T,,M,0.00,N,0.0,K,A*32<CR><LF>

Name	Example	Unit	Description
Message ID	$GPVTG		VTG protocol header
Course	83.37	degrees	Measured heading
Reference	T		True
Course		degrees	Measured heading

Reference	M		Magnetic1 [1]
Speed	0.00	knots	Measured horizontal speed
Units	N		Knots
Speed	0.0	km/hr	Measured horizontal speed
Units	K		Kilometers per hour
Mode	A		A=Autonomous D=DGPS
Checksum	*32		
<CR><LF>			End of message termination

NOTE

- Does not support magnetic declination. All “course over ground” data are geodetic WGS84 directions.

2.2.7 Message ID ZDA: Time & Data

Example:
$GPZDA,091926.000,17,12,2010,,*55<CR><LF>

Name	Example	Unit	Description
Message ID	$GPZDA		ZDA protocol header
UTC time	091926.000	Hhmmss.sss	The UTC time units are: hh = UTC hours from 00 to 23 mm = UTC minutes from 00 to 59 ss = UTC seconds from 00 to 59 sss= UTC micro seconds Either using valid IONO/UTC or estimated from default leap seconds
Day	17		Day of the month, range 1 to 31
Month	12		Month of the year, range 1 to 12
Year	2010		1980 to 2079
Local zone hour [1]		hour	Offset from UTC

Local zone minutes[1]		minute	Offset from UTC
Checksums	*55		
<CR><LF>			End of message termination

2.3 Proprietary NMEA Messages

PAIR command is an AIROHA proprietary GNSS data transferring protocol. This protocol is used to configure the GNSS module's parameters, to set/get aiding information, and to receive notifications from the GNSS module. To process data conveniently, the PAIR commands is aligned with the NMEA sentence format.

2.3.1 Packet Type:001 PAIR_ACK

Acknowledge of PAIR command

DataField: PAIR_ACK			
Name	**Unit**	**Default**	**Description**
Cmd	--	--	Command_ID: The command / packet type the acknowledge responds
Response Result	--	--	0 The command was successfully sent 1 The command is processing. You must wait for the result 2 Sending the command failed 3 This command ID is not supported 4 Command parameter error. Out of range / some parameters were lost / checksum error 5 MNL service is busy. You can try again soon

Return&Example

[Return]
$PAIR001,Command_ID,Result*CS<CR><LF>
Command_ID: The command / packet type the acknowledge responds
Result: The result of the command. The value is mnl_service_result_type_t
0: The command was successfully sent
1: The command is processing. You must wait for the result
2: Sending the command failed
3: This command ID is not supported
4: Command parameter error. Out of range / some parameters were lost /

checksum error

5: MNL service is busy. You can try again soon

[Example]

Send:

$PAIR666*3C\r\n

Response:

$ PAIR001,666,3*3E \r\n ==> $PAIR666 This command ID is not supported

NOTE

This item is the response of commands. The GNSS system automatically sends this command. Do not directly send it to the GNSS system.

2.3.2 Packet Type:002 PAIR_GNSS_SUBSYS_POWER_ON

Power on the GNSS system. Include DSP/RF/Clock and other GNSS modules.

DataField: $PAIR002*CS<CR><LF>

Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]

1. PAIR_ACK for send result

[Example]

Send:

$PAIR002*38\r\n

Response:

$PAIR001,002,1*38\r\n ==> The power on process is running. Please wait a moment.

$PAIR001,002,0*39\r\n ==> Power on was successful.

NOTE

Please send this command before using any location service.

2.3.3 Packet Type:003 PAIR_GNSS_SUBSYS_POWER_OFF

Power off GNSS system. Include DSP/RF/Clock and other GNSS modules.
CM4 also can receive commands (Include the AT command / the race Command / the part of PAIR command which is not dependent on DSP.) after sending this command.

DataField: $PAIR003*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR003*39\r\n
Response:
$PAIR001,003,1*39\r\n ==> The power off process is running. Please wait a moment.
$PAIR001,003,0*38\r\n ==> Power off was successful.

NOTE

The location service is not available after this command is executed.
The system can still receive configuration PAIR commands. The application is running if necessary.
CM4 will go to sleep if the application is not working at this time. The system can be awoken by the GNSS_DATA_IN_EINT pin after going to sleep.

2.3.4 Packet Type:004 PAIR_GNSS_SUBSYS_HOT_START

Hot Start. Use the available data in the NVRAM

DataField: $PAIR004*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR004*3E\r\n
Response:
$PAIR001,004,0*3F\r\n ==> Success

2.3.5 Packet Type:005 PAIR_GNSS_SUBSYS_WARM_START

Warm Start. Not using Ephemeris data at the start

DataField: $PAIR005*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR005*3F\r\n
Response:
$PAIR001,005,0*3E\r\n ==> Success

2.3.6 Packet Type:006 PAIR_GNSS_SUBSYS_COLD_START

Cold Start. Not using the Position, Almanac and Ephemeris data at the start

DataField: $PAIR006*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR006*3C\r\n
Response:
$PAIR001,006,0*3D\r\n ==> Success

2.3.7 Packet Type:007 PAIR_GNSS_SUBSYS_FULL_COLD_START

Full Cold Start
In addition to Cold start, this command clears the system/user configurations at the start
It resets the GNSS module to the factory default

DataField: $PAIR007*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR007*3D\r\n
Response:
$PAIR001,007,0*3C\r\n ==> Success

2.3.8 Packet Type:010 PAIR_REQUEST_AIDING

Request GNSS system reference data.
The GNSS system automatically sends this command. Please do not actively send it to the GNSS system.

Return&Example

[Return]

$PAIR010,<Type>,<GNSS_System>,<Week_Number>,<Time_of_Week>*CS<CR><LF>t.

Type: The data type.

0: Need to update EPO data.

1: Need to update the time.

2: Need to update the location.

GNSS_System: The GNSS system type is needed.

0: Need GPS data.

1: Need GLONASS data.

2: Need GALILEO data.

3: Need BEIDOU data.

4: Need QZSS data.

Week_Number: The current GNSS week number.

Time_of_Week: The current GNSS time of week.

[Example]

Response:

$PAIR010,0,0,2044,369413*33\r\n ==> Please send GPS EPO data when this command is received.

$PAIR010,1,-1*16\r\n ==> Please send reference time when this command is received.

$PAIR010,2,-1*15\r\n ==> Please send reference location when this command is received.

NOTE

The GNSS system automatically sends this command. Please do not actively send it to the GNSS system.

2.3.9 Packet Type:011 PAIR_INDICATION_SYSTEM_MESSAGE

GNSS System message indication

DataField: $PAIR011,<Type>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Type	--	--	The system message type "1", Notification for GNSS system startup

Return&Example

[Return]

NONE

[Example]

$PAIR011,001*27

NOTE

The GNSS system automatically sends this command. Please do not actively send it to the GNSS system.

2.3.10 Packet Type:012 PAIR_INDICATION_SYSTEM_WAKEUP

CM4 system wake up indication.
CM4 will entry sleep if application not working.
System can wake up by GNSS_DATA_IN_EINT Pin after entering sleep.
Application (gnss_demo project) need send this command as ACK to host after wakeup done.

DataField: $PAIR012*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
NONE

NOTE

The application (gnss_demo project) automatically sends this command. Please do not actively send it to the application.

2.3.11 Packet Type:020 PAIR_GET_VERSION

Query the firmware release information

DataField: $PAIR020*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**

--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR020,<Project Version>,<Frequency>,<SW package>,<Service version>,<Service build time>,
<DSP L1 rom version>,<DSP L1 ram version>,<DSP L5 rom version>,<DSP L5 ram version>,
<Kernel version>,<Kernel build time>,<KF version>,<KF build time>,
<RTK version>,<RTK build time>*CS<CR><LF>

Project Version:
<Project_board>_<SDK version>_<SDK Build time>
<Project_board> AG3335A / AG3335M / AG3335S/ AG3352Q
<SDK version> VX.Y.Z - X:Major Y:Minor Z. Bug fix
<SDK build time> YYYYMMDD
Ex:
AG3335A_V1.0.0_20190729

Frequency:
S: single
D: dual

SW package:
N: normal
W: raw
R: RTK
I: NavIC

Service version:
mnl_service version in 7 characters
Ex:
xxxxxxx

Service build time:
mnl_service library build time
Ex:
yyMMDDhhmm

DSP L1 rom version:
Null before first power on
Ex:
xx

DSP L1 ram version:
Null before first power on
Ex:
xxx

DSP L5 rom version:
Null for L1 only project

Null before first power on
Ex:
xx
DSP L5 ram version:
Null for L1 only project
Null before first power on
Ex:
xxx
Kernel version:
mnl_kernel version in 7 characters
Ex:
xxxxxxx
Kernel build time:
mnl_kernel library build time
Ex:
yyMMDDhhmm
KF version:
mnl_kf version in 7 characters
Ex:
xxxxxxx
KF build time:
mnl_kf library build time
Ex:
yyMMDDhhmm
RTK version:
RTK version in 7 characters
anything other than the RTK project
Ex:
xxxxxxx
RTK build time:
RTK library build time
Null for not RTK project
Ex:
yyMMDDhhmm

[Example]

Send:

$PAIR020*38\r\n

Response:

$PAIR001,020,0*39\r\n ==> Success

$PAIR020,AG3352Q_V2.1.0.AG3352_20220530,S,N,a94c01d,2204261517,2a7,0,,,e7ae9b61,2204261515,d01b1f2,2204261517,,*41\r\n

2.3.12 Packet Type:021 PAIR_GET_SETTING_INFO

Query the customer related setting, such as the firmware release information, DCB values, HW interface, ULP enable and NVRAM auto saving.

DataField: $PAIR021*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]

1. PAIR_ACK for send result

2. $PAIR021,<Project Version>,<Frequency>,<SW package>,<Service version>,<Service build time>,
<DSP L1 rom version>,<DSP L1 ram version>,<DSP L5 rom version>,<DSP L5 ram version>,
<Kernel version>,<Kernel build time>,<KF version>,<KF build time>,<RTK version>,
<RTK build time>,<GPS DCB>,<GAL DCB>,<BDS DCB>,<QZS DCB>,<TCXO Freq Error>,<Gain>,<SWPRT Info>,
,<ULP enable>,<NVRAM Auto Saving>*CS<CR><LF>

Refer to PAIR020 (Project Version,Frequency,...,RTK build time)

GPS DCB : The Differential Code Biases value for GPS.

GAL DCB : The Differential Code Biases value for GAL.

BDS DCB : The Differential Code Biases value for BDS.

QZS DCB : The Differential Code Biases value for QZS.

TCXO Frequency error :
'0' 0.5ppm
'1' 1.0ppm
'2' 1.5ppm
'3' 2.0ppm

Gain : '0' High gain
'1' Low gain

SWPRT Check :
##: No Check
#P: Pass
#F: Fail

ULP enable :
'0' disable
'1' enable

NVRAM Auto Saving :
'0' Disable
'1' Enable

[Example]

Send:
$PAIR021*39\r\n
Response:
$PAIR001,021,0*38\r\n
$PAIR021,AG3352Q_V2.1.0.AG3352_20220530,S,N,a94c01d,2204261517,2a7,0,,,e7ae9b61,2204261515,d01b1f2,2204261517,,,-15.48,-15.48,-14.02,-15.48,0,1,##,0,0*62\r\n

2.3.13 Packet Type:022 PAIR_READY_TO_READ

Host system wake up notification.
Application (gnss_demo project) will pull high GNSS_NOTIFY_HOST_WAKEUP_PIN > 10ms when data is ready to send to wake up host application.
Please send this command as ACK to SIM65M after wakeup done.

DataField: $PAIR022,<GPIO_PIN>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
GPIO_PIN	--	24, GPIO24	the GPIO pin id which used to wakeup. This is a ID to identity different IO controllers.

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR022*3A\r\n
Response:
$PAIR001,022,0*3B\r\n

NOTE

There is no need to use this command, if the host does not enter sleep or HW not set the configuration of GNSS_NOTIFY_HOST_WAKEUP_PIN.

2.3.14 Packet Type:023 PAIR_SYSTEM_REBOOT

Reboot GNSS whole chip, including the GNSS submodule and other all CM4 modules.

DataField: $PAIR023*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR023*3B\r\n
Response:
Reboot directly. Without Response.

2.3.15 Packet Type:024 PAIR_GET_CHIP_VERSION

Query the chip version.

DataField: $PAIR024*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR024,<CHIP>*CS<CR><LF>
CHIP: The chip version (A, M, S, etc.)
[Example]
Send:
$PAIR024*3C\r\n
Response:
$PAIR001,024,0*3D\r\n ==> Success
$PAIR024,NAN*51\r\n ==> AG3352Q

2.3.16 Packet Type:030 PAIR_COMMON_GET_POS_XYZ

The WGS84 ECEF XYZ Cartesian Position vector (in meters) with an estimated 1-sigma accuracy

DataField: $PAIR030*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR030,<X>,<Y>,<Z>,<Acc>*CS<CR><LF>
X: WGS84 ECEF X Cartesian position (meters)
Y: WGS84 ECEF Y Cartesian position (meters)
Z: WGS84 ECEF Z Cartesian position (meters)
Acc: 3-dimensional position space 1-sigma accuracy estimate (in meters)

[Example]
Send:
$PAIR030*39\r\n
Response:
$PAIR001,030,0*38\r\n ==> Success
$PAIR030,-2984524.0,4966958.3,2656485.3,3.0*14\r\n ==> The WGS84 ECEF XYZ Cartesian Position

2.3.17 Packet Type:031 PAIR_COMMON_GET_VEL_XYZ

The WGS84 ECEF XYZ Cartesian velocity vector (m/s) with an estimated 1-sigma accuracy

DataField: $PAIR031*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR031,<VX>,<VY>,<VZ>,<Acc>*CS<CR><LF>

VX: WGS84 ECEF X Cartesian velocity vector (m/s).
VY: WGS84 ECEF Y Cartesian velocity vector (m/s).
VZ: WGS84 ECEF Z Cartesian velocity vector (m/s).
Acc: 3-dimensional speed 1-sigma accuracy (m/s)
[Example]
Send:
$PAIR031*38\r\n
Response:
$PAIR001,031,0*39\r\n ==> Success
$PAIR031,0.19,-0.07,-0.11,0.49*3A\r\n ==> The WGS84 ECEF XYZ Cartesian Velocity

2.3.18 Packet Type:032 PAIR_COMMON_GET_GNSS_SATS_USED

Get used satellites (by constellation) for positioning

DataField: $PAIR032,<System_ID>*CS<CR><LF>

Name	Unit	Default	Description
System_ID	--	--	The GNSS constellation 0: GPS L1/L5, QZSS L1/L5 1: GLONASS L1 2: Galileo E1/E5a 3: BeiDou B1/B2a 4: Not support 5: NavIC L5

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR032,<System_ID>,<Signal_ID>,<Num_SV>,<PRN1>,<PRN2>,<PRN3>,....*CS<CR><LF>
System_ID: The GNSS constellation
Signal_ID:
GPS/QZSS 0: L1, 1: L5
GLONASS 0: L1
Galileo 0: E1, 1: E5a
BeiDou 0: B1, 1: B2a
NavIC 0: L5
Num_SV : Number of used satellites
PRN : Prn of used satellite
[Example]

Send:
$PAIR032,0*27\r\n
Response:
$PAIR001,032,0*3A\r\n ==> Success
$PAIR032,0,0,5,1,5,6,8,9*3D\r\n ==> GPS used number L1: 5, prn: 1,5,6,8,9
$PAIR032,0,1,3,1,6,8*36\r\n ==> GPS used number L5: 3, prn: 1,6,8
Send:
$PAIR032,1*26\r\n
Response:
$PAIR001,032,0*3A\r\n ==> Success
$PAIR032,1,0,5,77,78,81,82,88,*2F\r\n ==> Glonass used number L1: 5, prn: 77,78,81,82,88

2.3.19 Packet Type:033 PAIR_COMMON_GET_GNSS_SATS_IN_VIEW_STATUS

Get PRN, elevation, azimuth, CNR for satellites in view (by constellation)\
Each sentence maximum contains 12 satellites information

DataField: $PAIR033,<System_ID>*CS<CR><LF>

Name	Unit	Default	Description
System_ID	--	--	The GNSS constellation 0: GPS L1/L5, QZSS L1/L5 1: GLONASS L1 2: Galileo E1/E5a 3: BeiDou B1/B2a 4: Not support 5: NavIC L5

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR033,<System_ID>,<Signal_ID>,<Total_sentence>,<Sentence_index>,<Num_SV>,<PRN1>,<Elev1>,<Azim1>,<CNR1>,....*CS<CR><LF>
System_ID: The GNSS constellation
Signal_ID:
GPS/QZSS 0: L1, 1: L5
GLONASS 0: L1
Galileo 0: E1, 1: E5a
BeiDou 0: B1, 1: B2a
NavIC 0: L5
Total_sentence : total sentences of satellite info with maximum information of 12 satellites

per sentence.
Sentence_index : index of satellite information sentence (start at 1)
Num_SV : Number of satellites in view.
PRN : Prn of satellite in view.
Elev : Elevation angle of satellite in view. (Degree)
Azim : Azimuth of satellite in view. (Degree, True)
CNR : Signal strength of satellite in view. (dB-Hz)
[Example]
Send:
$PAIR033,0*26\r\n
Response:
$PAIR001,033,0*3B\r\n ==> Success
$PAIR033,0,0,1,1,5,1,79,33,50.0,3,28,145,39.0,7,49,215,48.0,8,27,54,40.0,11,61,17,47.0*3E\r\n
==> GPS L1 sv info
$PAIR033,0,1,1,1,5,1,79,33,53.0,3,28,145,42.0,7,49,215,50.0,8,27,54,42.0,11,61,17,48.0*34\r\n
==> GPS L5 sv info

2.3.20 Packet Type:034 PAIR_COMMON_GET_DOP

Get the DOP (Dilution of Precision)

DataField: $PAIR034*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR034,<HDOP>,<VDOP>,<PDOP>,<TDOP>,<GDOP>*CS<CR><LF>
HDOP: Horizontal dilution of precision
VDOP: Vertical dilution of precision
PDOP: Position (3D) dilution of precision
TDOP: Time dilution of precision
GDOP: Geometric dilution of precision
[Example]
Send:
$PAIR034*3D\r\n
Response:
$PAIR001,034,0*3C\r\n ==> Success
$PAIR034,1.01,1.70,1.99,0.63,1.56*0F\r\n

==> HDOP: 1.01, VDOP: 1.70, PDOP: 1.99, TDOP: 0.63, GDOP: 1.56

2.3.21 Packet Type:035 PAIR_COMMON_GET_FIX_STATUS

Get fix type and fix mode

DataField: $PAIR035*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR035,<FIX_TYPE>,<FIX_MODE>*CS<CR><LF>
FIX_TYPE:
0: NONE
1: SINGLE
2: DGPS
3: Not support
4: RTK FIX
5: RTK FLOAT
6: Estimated
FIX_MODE:
0: NONE
1: 2D fix
2: 3D fix

[Example]
Send:
$PAIR035*3C\r\n
Response:
$PAIR001,035,0*3D\r\n ==> Success
$PAIR035,2,2*3C\r\n ==> position 3D fix with Differential GPS

2.3.22 Packet Type:036 PAIR_COMMON_GET_HEADING

Get heading

DataField: $PAIR036*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR036,<Heading>*CS<CR><LF>
Heading: Heading over ground, degrees True

[Example]
Send:
$PAIR036*3F\r\n
Response:
$PAIR001,036,0*3E\r\n ==> Success
$PAIR036,120.2*3C\r\n ==> Heading: 120.2 degrees

2.3.23 Packet Type:037 PAIR_COMMON_GET_GPS_DGPS_STATUS

Get GPS satellite correction usage status

DataField: $PAIR037*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR037,<Total_Num>,<GPS_PRN1>,<GPS_PRN2>...*CS<CR><LF>

[Example]
Send:
$PAIR037*3E\r\n
Response:
$PAIR001,037,0*3F\r\n ==> Success
$PAIR037,10,1,3,7,9,11,13,17,22,23,30*19\r\n ==> 10 DGPS GPS satellites

2.3.24 Packet Type:043 PAIR_COMMON_GET_TOW_WN

Get TOW (Time of week) and WN (Week number) information.

DataField: $PAIR043*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

```
[Return]
    1. PAIR_ACK for send result.
    2. $PAIR043,<TOW>,<WN>*CS<CR><LF>
    TOW: GNSS Time of week
    WN: GNSS Week number
[Example]
Send:
    $PAIR043*3D\r\n
Response:
    $PAIR001,043,0*3C\r\n
    $PAIR043,2065,394925.000*22\r\n
```

2.3.25 Packet Type:044 PAIR_COMMON_GET_TTICK

Get system timer tick (units: 1 millisecond) [Range: 0~2147483647].
The tick will wrap back after exceeding its max value.

DataField: $PAIR044*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

```
[Return]
    1. PAIR_ACK for send result.
    2. $PAIR043,<Tick>*CS<CR><LF>
    Tick: system timer tick. Units: 1 millisecond.
[Example]
```

Send:
$PAIR044*3A\r\n
Response:
$PAIR001,044,0*3B\r\n
$PAIR044,102819*15\r\n

2.3.26 Packet Type:050 PAIR_COMMON_SET_FIX_RATE

Set Position Fix Interval.
If set less than 1000 ms, ASCII NMEA will automatically increase the update interval in order to decrease IO throughput.
It will return false if the operating voltage setting is not correct.

DataField: $PAIR050,<Fix_Interval>*CS<CR><LF>

Name	Unit	Default	Description
Fix_Interval	msec	--	Fix_Interval: Position fix interval in milliseconds (ms). [Range: 100 ~ 1000]

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR050,1000*12\r\n
Response:
$PAIR001,050,0*3E\r\n ==> Success

NOTE

For SIM65M module, <Fix_Interval> parameter only support 1000 ms.

2.3.27 Packet Type:051 PAIR_COMMON_GET_FIX_RATE

Get Position Fix Interval.

DataField: $PAIR051*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Fix_Interval	msec	--	Fix_Interval: Position fix interval in milliseconds (ms). [Range: 100 ~ 1000]

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR050,<Fix_Interval>*CS<CR><LF>
Fix_Interval: Position fix interval in milliseconds (ms). [Range: 100 ~ 1000]
[Example]
Send:
$PAIR051*3E\r\n
Response:
$PAIR001,051,0*3F\r\n ==> Success
$PAIR051,1000*13\r\n

2.3.28 Packet Type:058 PAIR_COMMON_SET_MIN_SNR

Set the minimum SNR of used satellites

DataField: $PAIR058,<MIN_SNR>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
MIN_SNR	--	--	Minimum SNR threshold of used satellites. (Valid range: 9~37, default value: 9)

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR058,15*1F\r\n
==> Set the minimum SNR threshold to 15, the chip would not use the satellite which SNR is smaller than 15.
Response:
$PAIR001,058,0*36\r\n ==> Success

2.3.29 Packet Type:059 PAIR_COMMON_GET_MIN_SNR

Query the minimum SNR of used satellites

DataField: $PAIR059*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR059,<MIN_SNR>*CS<CR><LF>
MIN_SNR: Minimum SNR threshold of used satellites. (Valid range: 9~37, default value: 9)
[Example]
Send:
$PAIR059*36\r\n
Response:
$PAIR001,059,0*37\r\n ==> Success
$PAIR059,15*1E\r\n

2.3.30 Packet Type:060 PAIR_COMMON_SET_ESTIMATED_NUM

Set the number of estimated fixes when entering the tunnel

DataField: $PAIR060,<DR_LIMIT>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
DR_LIMIT	--	--	Number of estimated fix. (Valid range: 0~500, default value: 0)

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:

$PAIR060,0*20\r\n ==> Disable the estimated fix when entering the tunnel
Response:
$PAIR001,060,0*3D\r\n ==> Success
Send:
$PAIR060,3*23\r\n ==> Keep outputting 3 fix when entering the tunnel
Response:
$PAIR001,060,0*3D\r\n ==> Success

2.3.31 Packet Type:061 PAIR_COMMON_GET_ESTIMATED_NUM

Query the number of estimated fixes when entering the tunnel

DataField: $PAIR061*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR061,<DR_LIMIT>*CS<CR><LF>
DR_LIMIT: Number of estimated fix. (Valid range: 0~500, default value: 0)
[Example]
Send:
$PAIR061*3D\r\n
Response:
$PAIR001,061,0*3C\r\n ==> Success
$PAIR061,0*21\r\n ==> The user disabled the DR estimated fix

2.3.32 Packet Type:062 PAIR_COMMON_SET_NMEA_OUTPUT_RATE

Set the NMEA sentence output interval of corresponding NMEA type

DataField: $PAIR062,<Type>,<Output_Rate>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Type	--	--	NMEA Type: -1 Reset all sentence to default value

			0 NMEA_SEN_GGA, // GGA interval - GPS Fix Data 1 NMEA_SEN_GLL, // GLL interval - Geographic Position - Latitude longitude 2 NMEA_SEN_GSA, // GSA interval - GNSS DOPS and Active Satellites 3 NMEA_SEN_GSV, // GSV interval - GNSS Satellites in View 4 NMEA_SEN_RMC, // RMC interval - Recommended Minimum Specific GNSS Sentence 5 NMEA_SEN_VTG, // VTG interval - Course Over Ground and Ground Speed 6 NMEA_SEN_ZDA, // ZDA interval - Time & Date
Output_Rate	--	--	Output interval setting: 0 - Disabled or not supported sentence 1 - Output once every one position fix 2 - Output once every two position fixes 3 - Output once every three position fixes 4 - Output once every four position fixes 5 - Output once every five position fixes

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR062,0,3*SS\r\n
Response:
$PAIR001,062,0*3F\r\n ==> Success

2.3.33 Packet Type:063 PAIR_COMMON_GET_NMEA_OUTPUT_RATE

Get the NMEA sentence output interval of corresponding NMEA type

DataField: $PAIR063,<Type>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Type	--	--	NMEA Type: -1 return all sentence configuration 0 NMEA_SEN_GGA, // GGA interval - GPS Fix Data 1 NMEA_SEN_GLL, // GLL interval - Geographic Position - Latitude longitude

			2 NMEA_SEN_GSA, // GSA interval - GNSS DOPS and Active Satellites 3 NMEA_SEN_GSV, // GSV interval - GNSS Satellites in View 4 NMEA_SEN_RMC, // RMC interval - Recommended Minimum Specific GNSS Sentence 5 NMEA_SEN_VTG, // VTG interval - Course Over Ground and Ground Speed 6 NMEA_SEN_ZDA, // ZDA interval - Time & DatePosition - Latitude longitude 2 NMEA_SEN_GSA, // GSA interval - GNSS DOPS and Active Satellites 3 NMEA_SEN_GSV, // GSV interval - GNSS Satellites in View 4 NMEA_SEN_RMC, // RMC interval - Recommended Minimum Specific GNSS Sentence 5 NMEA_SEN_VTG, // VTG interval - Course Over Ground and Ground Speed 6 NMEA_SEN_ZDA, // ZDA interval - Time & Date

Return&Example

[Return]

1. PAIR_ACK for send result

2. $PAIR063,<Type>,<Output_Rate>*CS<CR><LF>

Type: NMEA Type

0 NMEA_SEN_GGA, // GGA interval - GPS Fix Data

1 NMEA_SEN_GLL, // GLL interval - Geographic Position - Latitude longitude

2 NMEA_SEN_GSA, // GSA interval - GNSS DOPS and Active Satellites

3 NMEA_SEN_GSV, // GSV interval - GNSS Satellites in View

4 NMEA_SEN_RMC, // RMC interval - Recommended Minimum Specific GNSS Sentence

5 NMEA_SEN_VTG, // VTG interval - Course Over Ground and Ground Speed

6 NMEA_SEN_ZDA, // ZDA interval - Time & Date

Output_Rate: Output interval setting

0 - Disabled or not supported sentence

1 - Output once every one position fix

2 - Output once every two position fixes

3 - Output once every three position fixes

4 - Output once every four position fixes

5 - Output once every five position fixes

[Example]

Send:

$PAIR063,0*23\r\n

Response:

$PAIR001,063,0*3E\r\n ==> Success
$PAIR063,0,3*3C\r\n

2.3.34 Packet Type:064 PAIR_COMMON_SET_HACC_LIMIT

Set horizontal accuracy mask. Range from 30m to 200m or 0. GPS only gets the fix when hacc value < mask.

DataField: $PAIR064,<HaccMask>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
HaccMask	--	--	30~200: enable hacc mask feature. (Units: meter) 0 [Default Value]: disable hacc mask feature

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR064,50*11\r\n
Response:
$PAIR001,064,0*39\r\n ==> Success

NOTE

1. If horizontal accuracy > HaccMask is in use. The GNSS system will not output NMEA sentences

2.3.35 Packet Type:065 PAIR_COMMON_GET_HACC_LIMIT

Query horizontal accuracy mask

DataField: $PAIR065*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
HaccMask	--	--	Query horizontal accuracy mask

Return&Example

```
[Return]
    1. PAIR_ACK for send result
    2. $PAIR065,<HaccMask>*CS<CR><LF>
    HaccMask:
        30~200: enable hacc mask feature.   (Units: meter)
        0 [Default Value]: disable hacc mask feature
[Example]
Send:
    $PAIR065*39\r\n
Response:
    $PAIR001,065,0*38\r\n ==> Success
    $PAIR065,50*10\r\n
```

2.3.36 Packet Type:066 PAIR_COMMON_SET_GNSS_SEARCH_MODE

Configure the receiver to start searching for satellites. The setting is available when the NVRAM data is valid.
The device restarts when it receives this command.
Abbreviation: (GPS: "G", GLONASS: "R", Galileo: "E", BeiDou: "B", NavIC, "I")
Support constellation in L1 package: G/ GR/ GE/ GB/ GREB
Support constellation in L1 + L5 package: GREB / GEB
Support constellation in L1 + NavIC package G/ I/ GEI/ GREB/ GREBI
QZSS is always switchable.

DataField:
$PAIR066,<GPS_Enabled>,<GLONASS_Enabled>,<Galileo_Enabled>,<BeiDou_Enabled>,<QZSS_Enabled>,<NavIC_Enabled>*CS<CR><LF>

Name	Unit	Default	Description
GPS_Enabled	--	--	"0", disable (DO NOT search GPS satellites). "1", search GPS satellites
GLONASS_Enabled	--	--	"0", disable (DO NOT search GLONASS satellites). "1", search GLONASS satellites.
Galileo_Enabled	--	--	"0", disable (DO NOT search Galileo satellites). "1", search Galileo satellites
BeiDou_Enabled	--	--	"0", disable (DO NOT search BeiDou satellites). "1", search BeiDou satellites
QZSS_Enabled	--	--	"0", disable (DO NOT search QZSS satellites). "1", search QZSS satellites
NavIC_Enabled	--	--	"0", disable (DO NOT search NavIC satellites). "1", search NavIC satellites

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR066,1,0,0,0,0,0*3B\r\n ==> Search GPS satellites only
Response:
$PAIR001,066,0*3B\r\n ==> Success
Send:
$PAIR066,1,1,1,1,1,0*3B\r\n ==> Search GPS, GLONASS, Galileo, BeiDou, QZSS satellites
Response:
$PAIR001,066,0*3B\r\n ==> Success
Send:
$PAIR066,1,1,0,0,0,0*3A\r\n ==> Search GPS and GLONASS satellites
Response:
$PAIR001,066,0*3B\r\n ==> Success

NOTE

For SIM65M:
L1 single frequency, supports 5 modes G/ GR/ GE/ GB/ GREB as follows:
PAIR066,1,0,0,0,0,0 GPS only
PAIR066,1,1,0,0,0,0 GPS+GLONASS
PAIR066,1,0,1,0,0,0 GPS+GALILEO
PAIR066,1,0,0,1,0,0 GPS+ BEIDOU
PAIR066,1,1,1,1,0,0 GPS+GLONASS+GALILEO+BEIDOU

QZSS is always switchable.

2.3.37 Packet Type:067 PAIR_COMMON_GET_GNSS_SEARCH_MODE

This command is to get GPS, GLONASS, Galileo, BeiDou, QZSS and NavIC search settings

DataField: $PAIR067*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR067,<GPS_Enabled>,<GLONASS_Enabled>,<Galileo_Enabled>,<BeiDou_Enabled>,<QZSS_Enabled>,<NavIC_Enabled>*CS<CR><LF>
GPS_Enabled:
"0", disable (DO NOT search GPS satellites)
"1", search GPS satellites.
GLONASS_Enabled:
"0", disable (DO NOT search GLONASS satellites)
"1", search GLONASS satellites.
Galileo_Enabled:
"0", disable (DO NOT search Galileo satellites)
"1", search Galileo satellites.
BeiDou_Enabled:
"0", disable (DO NOT search BeiDou satellites)
"1", search BeiDou satellites.
QZSS_Enabled:
"0", disable (DO NOT search QZSS satellites)
"1", search QZSS satellites.
NavIC_Enabled:
"0", disable (DO NOT search NavIC satellites)
"1", search NavIC satellites
[Example]
Send:
$PAIR067*3B\r\n
Response:
$PAIR001,067,0*3A\r\n ==> Success
$PAIR067,1,0,0,0,0,0*3A\r\n ==> Search GPS satellites only

2.3.38 Packet Type:068 PAIR_COMMON_SET_HDOP_THRESHOLD

This command is for setting the HDOP threshold
If the HDOP value is larger than this threshold value, the position will not be fixed

DataField: $PAIR068,<HDOPThreshold>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
HDOPThreshold	--	--	"0": Disable this function Other value: Enable setting the HDOP threshold [Range:]

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR068,0.8*3E\r\n
Response:
$PAIR001,068,0*35\r\n ==> Success

2.3.39 Packet Type:069 PAIR_COMMON_GET_HDOP_THRESHOLD

This command is to get the HDOP threshold

DataField: $PAIR069*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
HDOPThreshold	--	--	0 Disable this function Other value Enable

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR069,<HDOPThreshold>*CS<CR><LF>
HDOPThreshold:
"0": Disable this function
Other value: Enable setting the HDOP threshold [Range:]
[Example]
Send:
$PAIR069*35\r\n
Response:
$PAIR001,069,0*34\r\n ==> Success
$PAIR069,0.8*3F\r\n

2.3.40 Packet Type:071 PAIR_COMMON_GET_STATIC_THRESHOLD

Query the static navigation speed threshold.

DataField: $PAIR071*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR071,<Speed_threshold>*CS<CR><LF>
Speed_threshold. 0~2 m/s. Default value is 0 m/s.
The minimum is 0.1 m/s, the maximum is 2.0 m/s.
[Example]
Send:
$PAIR071*3C\r\n
Response:
$PAIR001,071,0*3D\r\n ==> Success
$PAIR071,0.4*3A\r\n

2.3.41 Packet Type:070 PAIR_COMMON_SET_STATIC_THRESHOLD

Set the speed threshold for static navigation
If the actual speed is less than the threshold, the output position remains the same and the output speed will be zero
If the threshold value is set to 0, this function is disabled

DataField: $PAIR070,<Speed_threshold>*CS<CR><LF>			
Name	Unit	Default	Description
Speed_threshold	dm/s	--	0~20 dm/s. Default value is 0 dm/s The minimum is 1 dm/s, the maximum is 20 dm/s 1 dm/s = 0.1m/s

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR070,4*25\r\n
Response:

$PAIR001,070,0*3C\r\n ==> Success

2.3.42 Packet Type:072 PAIR_COMMON_SET_ELEV_MASK

Set satellite elevation mask
Satellites below the elevation mask are not used

DataField: $PAIR072,<Degree>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Degree	--	--	Satellite elevation-mask. (Valid range: -90 ~ 90, default value: 5)

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR072,5*26\r\n
Response:
$PAIR001,072,0*3E\r\n ==> Success

2.3.43 Packet Type:073 PAIR_COMMON_GET_ELEV_MASK

Get satellite elevation mask

DataField: $PAIR073*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR073,<Degree>*CS<CR><LF>
Degree: Satellite elevation-mask. (Valid range: -90 ~ 90, default value: 5)

[Example]
Send:
$PAIR073*3E\r\n
Response:
$PAIR001,073,0*3F\r\n ==> Success
$PAIR073,5*27\r\n

2.3.44 Packet Type:074 PAIR_COMMON_SET_AIC_ENABLE

Enable or disable active interference cancellation function

DataField: $PAIR074,<Enabled>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enabled	--	--	Enable or disable: '0' = Disable '1' = Enable (Default)

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR074,1*24\r\n
Response:
$PAIR001,074,0*38\r\n ==> Success

2.3.45 Packet Type:075 PAIR_COMMON_GET_AIC_STATUS

Get the status of active interference cancellation function.

DataField: $PAIR075*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR075,<Enabled>*CS<CR><LF>
Enabled: Enable or disable
"0", Disable.
"1", Enable.
[Example]
Send:
$PAIR075*38\r\n
Response:
$PAIR001,075,0*39\r\n ==> Success
$PAIR075,0*24\r\n ==> AIC is disabled.

2.3.46 Packet Type:076 PAIR_COMMON_SET_DATUM

Set default datum

DataField: $PAIR076,<Datum>*CS<CR><LF>

Name	Unit	Default	Description
Datum	--	--	0: WGS84 1: TOKYO-M 2: TOKYO-A ...

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR076,0*27\r\n
Response:
$PAIR001,076,0*3A\r\n ==> Success

NOTE

The total datums list in the AppendixC Datum List

2.3.47 Packet Type:077 PAIR_COMMON_GET_DATUM

Get default datum

DataField: $PAIR077*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR077,<Datum>*CS<CR><LF>
Datum:
0: WGS84 [Default]
1: TOKYO-M
2: TOKYO-A
The total datum list in the Appendix C Datum List
[Example]
Send:
$PAIR077*3A\r\n
Response:
$PAIR001,077,0*3B\r\n ==> Success
$PAIR077,0*26\r\n

2.3.48 Packet Type:078 PAIR_COMMON_SET_DATUM_ADVANCE

Set user-defined datum

DataField: $PAIR078,<majA>,<ecc>,<dX>,<dY>,<dZ>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
majA	m	--	User defined datum semi-major axis [m] [Range: 0 ~ 7000000]
ecc	m	--	User defined datum eccentric [m] [Range: 0 ~ 330]
dX	m	--	User defined datum to WGS84 X axis offset [m]
dY	m	--	User defined datum to WGS84 X axis offset [m]
dZ	m	--	User defined datum to WGS84 X axis offset [m]

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR078,6377397.155,299.1528128,-148.0,507.0,685.0*10\r\n
Response:
$PAIR001,078,0*34\r\n ==> Success

2.3.49 Packet Type:079 PAIR_COMMON_GET_DATUM_ADVANCE

Get user-defined datum

DataField: $PAIR079*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR079,<majA>,<ecc>,<dX>,<dY>,<dZ>*CS<CR><LF>
majA: User defined datum semi-major axis [m] [Range: 0 ~ 7000000]
ecc: User defined datum eccentric [m] [Range: 0 ~ 330]
dX: User defined datum to WGS84 X axis offset [m]
dY: User defined datum to WGS84 X axis offset [m]
dZ: User defined datum to WGS84 X axis offset [m]
[Example]
Send:
$PAIR079*34\r\n
Response:
$PAIR001,079,0*35\r\n ==> Success
$PAIR079,6377397.155, 299.1528128, -148.0, 507.0,685.0*31\r\n

2.3.50 Packet Type:080 PAIR_COMMON_SET_NAVIGATION_MODE

Set navigation mode

DataField: $PAIR080,<CmdType>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
CmdType	--	--	'0' Normal mode: For general purpose '1' [Default Value] Fitness mode: For running and walking activities so that the low-speed (< 5 m/s) movement will have more of an effect on the position calculation. '2' Reserved '3' Reserved '4' Stationary mode: For stationary applications where a zero dynamic assumed. '5' Reserved '6' Reserved '7' Swimming mode: For swimming purpose so that it smooths the trajectory and improves the accuracy of distance calculation.

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR080,1*2F\r\n ==> Enter fitness mode
Response:
$PAIR001,080,0*33\r\n ==> Success

2.3.51 Packet Type:081 PAIR_COMMON_GET_NAVIGATION_MODE

Get navigation mode

DataField: $PAIR081*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR081,<CmdType>*CS<CR><LF>
CmdType:
'0' [Default Value] Normal mode: For general purpose
'1' Fitness mode: For running and walking activities so that the low-speed (< 5 m/s) movement will have more of an effect on the position calculation.
'2' Reserved
'3' Reserved
'4' Stationary mode: For stationary applications where a zero dynamic assumed.
'5' Reserved
'6' Reserved
'7' Swimming mode: For swimming purpose so that it smooths the trajectory and improves the accuracy of distance calculation.

[Example]
Send:
$PAIR081*33\r\n
Response:
$PAIR001,081,0*32\r\n ==> Success
$PAIR081,1*2E\r\n ==> Current is fitness mode.

2.3.52 Packet Type:083 PAIR_COMMON_GET_HIGH_SENSITIVITY_TRACKING_MODE

Query setting of position output disabled/enabled in high-sensitivity tracking mode

DataField: $PAIR083*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR083,<Status>*CS<CR><LF>
0: Enable, 1: Disable

[Example]
Send:
$PAIR083*31\r\n
Response:

$PAIR001,083,0*30\r\n ==> Success
$PAIR083,0*2D\r\n ==> Enable high sensitivity tracking mode. GNSS system will get fix in high sensitivity tracking

2.3.53 Packet Type:086 PAIR_COMMON_SET_DEBUGLOG_OUTPUT

This command is to set enable/disable debug log output in binary format

DataField: $PAIR086,<Status>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Status	--	--	0: Disable 1: Enable with full debuglog output 2: Enable with lite debuglog output

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR086,1*29\r\n
Response:
$PAIR001,086,0*35\r\n ==> Success

2.3.54 Packet Type:087 PAIR_COMMON_GET_DEBUGLOG_OUTPUT

Query setting of debug log output.

DataField: $PAIR087*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.

2. $PAIR087,<Status>*CS<CR><LF>
   0: Disable
   1: Enable with full debuglog output
   2: Enable with lite debuglog output

[Example]
Send:
$PAIR087*35\r\n
Response:
$PAIR001,087,0*34\r\n ==> Success
$PAIR087,1*28 ==> Enable Debuglog output

2.3.55 Packet Type:090 PAIR_COMMON_SET_QUICKQR_ENABLE

Enable quick QR mode

DataField: $PAIR090,<Enable>*CS<CR><LF>			
Name	Unit	Default	Description
Enable	--	--	0: disable quick QR mode 1: enable quick QR mode

Return&Example

[Return]
1. PAIR_ACK for send result.

[Example]
Send:
$PAIR090,0*2F\r\n ==> Disable Quick QR mode
Response:
$PAIR001,090,0*32\r\n ==> Success

2.3.56 Packet Type:091 PAIR_COMMON_GET_QUICKQR_STATUS

Query current quick QR mode

DataField: $PAIR091*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR091,<Enable>*CS<CR><LF>
Enable:
0: disable quick QR mode
1: enable quick QR mode
[Example]
Send:
$PAIR091*32\r\n
Response:
$PAIR001,091,0*33\r\n ==> Success
$PAIR091,1*2F\r\n ==> In Quick QR mode

2.3.57 Packet Type:092 PAIR_COMMON_SET_STATIC_MODE

Enable static mode

DataField: $PAIR092,<Enable>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	0: Disable static mode 1: Enable static mode (Default)

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR092,1*2C\r\n
Response:
$PAIR001,092,0*30\r\n ==> Success

2.3.58 Packet Type:093 PAIR_COMMON_GET_STATIC_MODE

Query if current mode is static mode

DataField: $PAIR093*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR093,<Enable>*CS<CR><LF>
Enable:
0: Not in static mode
1: In static mode
[Example]
Send:
$PAIR093*30\r\n
Response:
$PAIR001,093,0*31\r\n ==> Success
$PAIR093,1*2D\r\n ==> In static mode

2.3.59 Packet Type:098 PAIR_COMMON_SET_NMEA_POS_DECIMAL_PRECISION

This command is for setting the digits shown in the NMEA position

DataField: $PAIR098,<MODE>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
MODE	--	--	0: Latitude, Longitude in 4 digits, Altitude in 1 digit 1: Latitude, Longitude in 5 digits, Altitude in 2 digit 2: Latitude, Longitude in 6 digits, Altitude in 3 digit 3: Latitude, Longitude in 7 digits, Altitude in 3 digit

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR098,0*27\r\n

==> Set the Lat/Lon digit 4 digit, and Alt in 1 digit (GGA/GLL/RMC)
Response:
$PAIR001,098,0*3A\r\n ==> Success

2.3.60 Packet Type:099 PAIR_COMMON_GET_NMEA_POS_DECIMAL_PRECISION

This command is to get NMEA position shown digit mode

DataField: $PAIR098,<MODE>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR099,<MODE>*CS<CR><LF>
MODE:
0: Latitude, Longitude in 4 digits, Altitude in 1 digit (Default)
1: Latitude, Longitude in 5 digits, Altitude in 2 digit
2: Latitude, Longitude in 6 digits, Altitude in 3 digit
3: Latitude, Longitude in 7 digits, Altitude in 3 digit
[Example]
Send:
$PAIR099*3A\r\n
Response:
$PAIR001,099,0*3B\r\n ==> Success
$PAIR099,0*26\r\n ==> Latitude, Longitude in 4 digits, Altitude in 1 digit

2.3.61 Packet Type:100 PAIR_COMMON_SET_NMEA_OUTPUT_MODE

This command is to set NMEA output mode

DataField: $PAIR100,<NMEA_MODE>,<PROPRIETARY_MODE>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
NMEA_MODE	--	--	0: Disable NMEA 1: ASCII NMEA v4.1(Default)

			2: ASCII NMEA v3.0
PROPRIETARY_MODE	--	--	0: Disable extra proprietary sentence (Default) 1: Enable proprietary sentence

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR100,1,0*3A\r\n ==> ASCII NMEA v4.1, Disable extra proprietary sentence
Response:
$PAIR001,100,0*3A\r\n ==> Success
Send:
$PAIR100,0,1*3A\r\n ==> No ASCII NMEA output, Enable proprietary sentence
Response:
$PAIR001,100,0*3A\r\n ==> Success

2.3.62 Packet Type:101 PAIR_COMMON_GET_NMEA_OUTPUT_MODE

This command is to get NMEA output mode

DataField: $PAIR101,*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR101,<NMEA_MODE>,<PROPRIETARY_MODE>*CS<CR><LF>
NMEA_MODE:
0: Disable NMEA
1: ASCII NMEA v4.1(Default)
2: ASCII NMEA v3.0
PROPRIETARY_MODE:
0: Disable extra proprietary sentence (Default)
1: Enable proprietary sentence
[Example]
Send:
$PAIR101*3A\r\n
Response:

$PAIR001,101,0*3B\r\n ==> Success
$PAIR101,0,1*3B\r\n ==> Disable NMEA output, enable proprietary sentence

2.3.63 Packet Type:106 PAIR_COMMON_SET_CPU_FREQ_LEVEL

This command is to set the CPU frequency level. Only work when GNSS is power on.
(It returns an error when user enables ULP)

DataField: $PAIR106,<Level>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Level	--	--	0: change to normal CPU frequency 1: change to high CPU frequency

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR106,1*20\r\n
Response:
$PAIR001,106,0*3C\r\n ==> Success

2.3.64 Packet Type:107 PAIR_COMMON_GET_CPU_FREQ_LEVEL

This command is to get current CPU frequency level

DataField: $PAIR105,*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR107,<Level>*CS<CR><LF>
Level:

0: The CPU frequency is normal level.
1: The CPU frequency is high level.
[Example]
Send:
$PAIR107*3C\r\n
Response:
$PAIR001,107,0*3D\r\n ==> Success
$PAIR107,0*20\r\n ==> Normal CPU frequency level

2.3.65 Packet Type:120 PAIR_COMMON_SET_PROPRIETARY_OUTPUT_RATE

Set the proprietary message output interval of the corresponding ascii/binary proprietary type.

Please refer to "Airoha_IoT_SDK_for_GNSS_Developers_Guide" for information about the proprietary type.

Note: You can only configure output rate which the mode you set in gnss_config.bin by configuration tool.

DataField: $PAIR120,<Mode>,<Type>,<Output_Rate>*CS<CR><LF>

Name	Unit	Default	Description
Mode	--	--	0: ASCII proprietary mode 1: Binary proprietary mode
Type	--	-1	-1 Reset all messages to the default value. When Mode is ASCII: 0: PAIRDGP 1: PLSV 2: GPACCURACY 3: reserved 4: EPE 5: reserved 6: PAIRSAT When Mode is Binary: 0: PAIRDGP 1: reserved 2: reserved 3: reserved 4: EPE 6: PAIRSAT 7: PVT 8: PVT additional 9: SV status
Output_Rate	--	1	Output interval setting (Valid range: 0~20) 0 - Disabled or not supported sentence

			1 - Output once every one position fix 2 - Output once every two position fixes 3 - Output once every three position fixes 4 - Output once every four position fixes 5 - Output once every five position fixes

Return&Example

```
[Return]
    1. PAIR_ACK for send result
[Example]
Send:
      $PAIR120,0,2,3*24\r\n
Response:
      $PAIR001,120,0*38\r\n ==> Success
```

2.3.66 Packet Type:121 PAIR_COMMON_GET_PROPRIETARY_OUTPUT_RATE

Get the proprietary message output interval of the corresponding proprietary type.

Please refer to "Airoha_IoT_SDK_for_GNSS_Developers_Guide" for information about the proprietary type.

Note: You can only configure output rate which the mode you set in gnss_config.bin by configuration tool.

DataField: $PAIR121,<Mode>,<Type>*CS<CR><LF>

Name	Unit	Default	Description
Mode	--	--	0: ASCII proprietary mode 1: Binary proprietary mode
Type		--	-1 Return all sentence configurations. When Mode is ASCII: 0: PAIRDGP 1: PLSV 2: GPACCURACY 3: reserved 4: EPE 5: reserved 6: PAIRSAT When Mode is Binary: 0: PAIRDGP 1: reserved 2: reserved 3: reserved

			4: EPE 6: PAIRSAT 7: PVT 8: PVT additional 9: SV status

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR121,<Mode>,<Type>,<Output_Rate>*CS<CR><LF>
Mode:
0: ASCII proprietary mode
1: Binary proprietary mode
Type:
-1 Return all sentence configurations.
When Mode is ASCII:
0: PAIRDGP
1: PLSV
2: GPACCURACY
3: reserved
4: EPE
5: reserved
6: PAIRSAT
When Mode is Binary:
0: PAIRDGP
1: reserved
2: reserved
3: reserved
4: EPE
6: PAIRSAT
7: PVT
8: PVT additional
9: SV status
Output_Rate: Output interval setting (Valid range: 0~20, default value: 1)
0 - Disabled or not supported sentence
1 - Output once every one position fix
2 - Output once every two position fixes
3 - Output once every three position fixes
4 - Output once every four position fixes
5 - Output once every five position fixes

[Example]

Send:

$PAIR121,0,2*3A\r\n

Response:

```
$PAIR001,121,0*39\r\n ==> Success
$PAIR121,0,2,3*25\r\n
```

2.3.67 Packet Type:123 PAIR_SIMCOM_VERSION

Query the release version of simcom

DataField: $PAIR123*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
	--	--	--

Return&Example

[Return]

1. PAIR_ACK for send result

2. 2. $PAIR123,<Simcom Release Version>

[Example]

Send:

$PAIR123*3A\r\n

Response:

$PAIR001,123,0*3B

$PAIR123,B01V02SIM65M_11*47

NOTE

For SIM65M module, firmware version no less than B01V03SIM65M_11: command ”**PAIR123**” is replaced by “**PAIR10001**”.

2.3.68 Packet Type:126 PAIR_COMMON_SET_BD_GEO_ENABLE

Enable tracking BeiDou GEO satellite.

DataField: $PAIR126,<Enable>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	0: Disable

			1: Enable (Default)

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR126,0*23\r\n ==> Disable tracking BeiDou GEO satellite
Response:
$PAIR001,126,0*3E\r\n ==> Success

2.3.69 Packet Type:127 PAIR_COMMON_GET_BD_GEO_ENABLE

Query if tracking Beidou GEO satellite is enabled.

DataField: $PAIR127,*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR127,<Enable>*CS<CR><LF>
Enable:
0: Disable
1: Enable
[Example]
Send:
$PAIR127*3E\r\n
Response:
$PAIR001,127,0*3F\r\n ==> Success
$PAIR127,0*22\r\n ==> Tracking Beidou GEO satellite is disabled

2.3.70 Packet Type:130 PAIR_COMMON_SET_SV_BLACKLIST

Set sv blacklist for selected constellation, which excluding the specific PRNs and do not search them.

DataField: $PAIR130,<System_ID>,<Blacklist>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
System_ID	--	--	The GNSS constellation 0: GPS L1/L5 1: GLONASS L1 2: Galileo E1/E5a 3: BeiDou B1/B2a 4: QZSS L1/L5 5: NavIC L5
Blacklist			Bitwise format in HEX (LSB for QZSS NMEA PRN: 193, GLONASS NMEA PRN: 65, others PRN: 1)

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR130,0,8000C001*72\r\n ==> Disable tracking GPS PRN 1,15,16,32
Response:
$PAIR001,130,0*39\r\n ==> Success

2.3.71 Packet Type:131 PAIR_COMMON_GET_SV_BLACKLIST

Get sv blacklist for selected constellation, which excluding the specific PRNs and do not search them.

DataField: $PAIR131,<System_ID>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
System_ID	--	--	The GNSS constellation 0: GPS L1/L5 1: GLONASS L1 2: Galileo E1/E5a 3: BeiDou B1/B2a 4: QZSS L1/L5 5: NavIC L5

Return&Example

[Return]
1. PAIR_ACK for send result.

2. $PAIR131,<System_ID>,<Blacklist>*CS<CR><LF>
System_ID: The GNSS constellation
0: GPS L1/L5
1: GLONASS L1
2: Galileo E1/E5a
3: BeiDou B1/B2a
4: QZSS L1/L5
5: NavIC L5
Blacklist: Bitwise format in HEX
(LSB for QZSS NMEA PRN: 193, GLONASS NMEA PRN: 65, others PRN: 1)

[Example]
Send:
$PAIR131,0*25\r\n
Response:
$PAIR001,131,0*38\r\n ==> Success
$PAIR131,0,8000C001*73\r\n ==> Disable tracking GPS PRN 1,15,16,32

2.3.72 Packet Type:378 PAIR_TEST_INITIALIZE

Initialize for test mode. Test command must be sent after receiving the command success response.

DataField: $PAIR378*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.

[Example]
Send:
$PAIR378*36\r\n
Response:
$PAIR001,378,1*36\r\n ==> Processing
$PAIR001,378,0*37\r\n ==> Success

2.3.73 Packet Type:382 PAIR_TEST_LOCK_SYSTEM_SLEEP

Test command for lock system sleep.
CM4 will entry sleep if application not working. System can be wake up by GNSS_DATA_IN_EINT Pin

after entry sleep.

You can send this command to forbid/ permit sleep for special test scene.

DataField: $PAIR382,<Lock>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Lock	--	--	Lock sleep or not. 1, Lock sleep. 0: Unlock sleep.

Return&Example

[Return]

1. PAIR_ACK for send result

[Example]

Send:

$PAIR382,1*2E\r\n ==> Forbid sleep

Response:

$PAIR001,382,0*32\r\n ==> Forbid Sleep Success. System will not enter sleep.

2.3.74 Packet Type:383 PAIR_TEST_SEND_LOG

Test Command. Send log data to GNSS chip. This command will be saved to the log file.

DataField: $PAIR0383,<ref_loc_lat>,<ref_loc_lon>,<ref_loc_alt>,<ref_utc>,<ref_date>,<curr_loc_lat>,<curr_loc_lon>,<curr_loc_alt>,<curr_utc>,<curr_date>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
ref_loc_lat	--	--	Reference Latitude. Format is xxmm.dddd. (xx: degrees. mm: minutes. dddd: decimal part of minutes.)
ref_loc_lon			Reference Longitude. Format is xxmm.dddd. (xx: degrees. mm: minutes. dddd: decimal part of minutes.)
ref_loc_alt			Reference Altitude. Mean-sea-level (geoid). (Meters.)
ref_utc			Reference UTC time of position. Format is hhmmss.ddd. (hh: hours. mm: minutes. ss: seconds. ddd: decimal part of seconds.)
ref_date			Reference Date of position. Format is ddmmyy. (dd: data. mm: month.yy: year.)
curr_loc_lat			Latitude from NMEA. Format is xxmm.dddd. (xx: degrees. mm: minutes. dddd: decimal part of minutes.)
curr_loc_lon			Longitude from NMEA. Format is xxmm.dddd. (xx: degrees. mm: minutes. dddd: decimal part of minutes.)
curr_loc_alt			Altitude from NMEA. Mean-sea-level (geoid). (Meters.)
curr_utc			UTC time of position from NMEA. Format is hhmmss.ddd. (hh: hours. mm: minutes. ss: seconds. ddd: decimal part

			of seconds.)
curr_date			Date of position from NMEA. Format is ddmmyy. (dd: data. mm: month.yy: year.)

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR383,3032.4300,10403.7698,531.9,050919,090703,3032.4300,10403.7698,531.9,050919,090703*32\r\n
Response:
$PAIR001,383,0*33\r\n ==> Send Success

2.3.75 Packet Type:391 PAIR_TEST_JAMMING_DETECT

Jamming detection test command

DataField: $PAIR391, <CmdType>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
CmdType	--	--	"0" disable jamming detection message output. "1" enable jamming detection message output

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIRSPF,<Jamstatus>*CS<CR><LF>
Jamstatus: "0" Unknown Status.
"1" No jamming, healthy status.
"2" Warning status.
"3" Critical status. [Example]
Send:
$PAIR391,1*2C\r\n
Enable the jamming detection message output
Response:
$PAIR001,391,0*30\r\n ==> Success
$PAIRSPF,1*52\r\n ==> L1 band result

Send:
$PAIR391,0*2D\r\n
Disable the jamming detection message output
Response:
$PAIR001,391,0*30\r\n ==> Success

2.3.76 Packet Type:392 PAIR_TEST_JAMMING_SCAN

Jamming scan test command

DataField:
$PAIR392, <JamScanType>,<JamScanNum>,<GloSubChan>,<Resolution>*CS<CR><LF>

Name	Unit	Default	Description
JamScanType	--	--	'0' enable GPS L1 band jamming scan '1' enable GLONASS L1 band jamming scan '2' enable BeiDou L1 band jamming scan '3' enable L5 band jamming scan
JamScanNum	--	--	Jamming scan test times. [Range: 1~255]
GloSubChan	--	--	GLONASS sub channel
Resolution	--	--	Jamming scan frequency resolution (L1 band only support Legacy, L5 band only support 50Hz) '0' Legacy (21KHz~61KHz) '1' 50Hz

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR392,0,50,0,0*07\r\n
GPS L1 band jamming scan test 50 times
Response:
$PAIR001,392,0*33\r\n ==> Success

2.3.77 Packet Type:393 PAIR_TEST_CW_MODE

Test CW (Continuous Wave) mode, and report CNR of CW.

DataField: $PAIR393,<Enabled>,<Signal_type>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enabled	--	--	0 (Disable), 1 (Enable)
Signal_type	--	--	"1" L1: 1575.42 MHz "2" L5: 1177.42 MHz (only support A/M/SD Dual band project) "3" NavIC: 1176.92 MHz (only support NavIC project) (In NavIC project, GNSS system must be reset when switch L1 or NavIC CW test)

Return&Example

[Return]

1. PAIR_ACK for send result.

2. $PAIR393,<CNR>,<ClockDrift>*CS<CR><LF>

CNR: CNR of CW (unit: dB-Hz)

ClockDrift: Clock drift (unit: ppm)

L5 only return the CNR since it has the same clock source as L1

[Example]

Send:

$PAIR393,1,1*33\r\n

CW Test L1 signal path

Response:

$PAIR001,393,0*32\r\n ==> Success

$PAIR393,0050,-0.125*33\r\n

Send:

$PAIR393,1,2*30\r\n

CW Test L5 signal path

Response:

$PAIR001,393,0*32\r\n ==> Success

$PAIR393,0050*1A\r\n

Send:

$PAIR393,1,3*31\r\n

CW Test NavIC signal path

Response:

$PAIR001,393,0*32\r\n ==> Success

$PAIR393,0050,-0.125*33\r\n

2.3.78 Packet Type:400 PAIR_DGPS_SET_MODE

DGPS correction data source mode

DataField: $PAIR400,<Mode> *CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Mode	--	--	DGPS data source mode: '0': No DGPS source '1': RTCM '2': SBAS(Include WAAS/EGNOS/GAGAN/MSAS) (Default) '3': SLAS

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR400,2*20\r\n ==> Set SBAS Mode
Response:
$PAIR001,400,0*3F\r\n ==> Success

2.3.79 Packet Type:401 PAIR_DGPS_GET_MODE

Query the DGPS data source mode

DataField: $PAIR401*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR401,<Mode>*CS<CR><LF>
Mode: DGPS data source mode.
'0': No DGPS source

```
        '1': RTCM
        '2': SBAS(Include WAAS/EGNOS/GAGAN/MSAS)
        '3': SLAS [Example]
Send:
    $PAIR401*3F\r\n
Response:
    $PAIR001,401,0*3E\r\n ==> Success
    $PAIR401,2*21\r\n   ==> SBAS Mode
```

2.3.80 Packet Type:410 PAIR_SBAS_ENABLE

Enable searching a SBAS satellite or not.
When navigation mode is Fitness or Swimming mode, SBAS is not supported.

DataField: $PAIR410,<Enabled>*CS<CR><LF>

Name	Unit	Default	Description
Enabled	--	--	Enable or disable: '0' = Disable '1' = Enable (Default)

Return&Example

```
[Return]
    1. PAIR_ACK for send result
[Example]
Send:
    $PAIR410,1*22\r\n ==> Enable SBAS
Response:
    $PAIR001,410,0*3E\r\n ==> Success
```

2.3.81 Packet Type:411 PAIR_SBAS_GET_STATUS

Query the status of SBAS to whether it is enabled.

DataField: $PAIR411*CS<CR><LF>

Name	Unit	Default	Description
--	--	--	--

Return&Example

```
[Return]
    1. PAIR_ACK for send result
    2. $PAIR411,<Enabled>*CS<CR><LF>
    Enabled: Enable or disable
    '0' = Disable
    '1' = Enable
[Example]
Send:
     $PAIR411*3E\r\n
Response:
     $PAIR001,411,0*3F\r\n ==> Success
     $PAIR411,1*23\r\n ==> Enable SBAS
```

2.3.82 Packet Type:412 PAIR_SBAS_GET_SAT_INFO

Get information about the SBAS satellites, such as SVid, SNR, azimuth, and elevation.

DataField: $PAIR412*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

```
[Return]
    1. PAIR_ACK for send result
    2. $PAIR412,<SVid>,<SNR>,<Azim>,<Elev>*CS<CR><LF>
[Example]
Send:
     $PAIR412*3D\r\n
Response:
     $PAIR001,412,0*3C\r\n ==> Success
     $PAIR412,50,42,134,50*0D\r\n
```

2.3.83 Packet Type:420 PAIR_SLAS_ENABLE

Enable the QZSS SLAS (Sub-meter Level Augmentation Service) operation.

DataField: $PAIR420,<Enabled>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enabled	--	--	'0' = Disable (Default) '1' = Enable

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR420,1*21\0d\0a ==> Enable QZSS SLAS
Response:
$PAIR001,420,0*3D\0d\0a ==> Success

2.3.84 Packet Type:421 PAIR_SLAS_GET_STATUS

Query the status of SLAS to check whether it is enabled.

DataField: $PAIR421*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR421,<Enabled>*CS<CR><LF>
Enabled: Enable or disable
'0' = Disable
'1' = Enable
[Example]
Send:
$PAIR421*3D\0d\0a
Response:
$PAIR001,421,0*3C\0d\0a ==> Success
$PAIR421,1*20\0d\0a ==> The status of QZSS SLAS is enabled

2.3.85 Packet Type:430 PAIR_RTCM_SET_INPUT_VERSION

Set RTCM input version.

DataField: $PAIR430,<Version>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Version	--	--	Input version 0: RTCM v2.x (Default) 1: RTCM v3.x

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR430,0*21\r\n ==> set RTCM v2.x input
Response:
$PAIR001,430,0*3C\r\n ==> Success

2.3.86 Packet Type:431 PAIR_RTCM_GET_INPUT_VERSION

Get RTCM input version.

DataField: $PAIR431*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR431,<Version>*CS<CR><LF>
Version: Input version
0: RTCM v2.x
1: RTCM v3.x
[Example]

Send:
$PAIR431*3C\r\n
Response:
$PAIR001,431,0*3D\r\n ==> Success
$PAIR431,0*20\r\n ==> RTCM v2.x input

2.3.87 Packet Type:432 PAIR_RTCM_SET_OUTPUT_MODE

Set RTCM output mode.

DataField: $PAIR432,<MODE>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
MODE	--	--	Measurement output mode (MSM4/MSM7) -1: Output None (Default) 0: Output RTCM3.x with message type MSM4 1: Output RTCM3.x with message type MSM7

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR432,1*22\r\n ==> set RTCM3.x output with type MSM7
Response:
$PAIR001,432,0*3E\r\n ==> Success

2.3.88 Packet Type:433 PAIR_RTCM_GET_OUTPUT_MODE

Get RTCM output mode.

DataField: $PAIR433*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR433,<MODE>*CS<CR><LF>
MODE: Measurement output mode (MSM4/MSM7)
-1: Output None (Default)
0: Output RTCM3.x with message type MSM4
1: Output RTCM3.x with message type MSM7
[Example]
Send:
$PAIR433*3E\r\n
Response:
$PAIR001,433,0*3F\r\n ==> Success
$PAIR433,0*22\r\n ==> RTCM3.x output with type MSM4

2.3.89 Packet Type:434 PAIR_RTCM_SET_OUTPUT_ANT_PNT

This command is to set enable/disable stationary antenna reference point for RTCM output.

DataField: $PAIR434,<ENABLE>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
ENABLE	--	--	Stationary antenna reference point (Message type 1005) 0: Disable 1: Enable

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR434,1*24\r\n ==> set RTCM3.x output with message type 1005
Response:
$PAIR001,434,0*38\r\n ==> Success

2.3.90 Packet Type:435 PAIR_RTCM_GET_OUTPUT_ANT_PNT

Query setting of stationary antenna reference point for RTCM output.

DataField: $PAIR435*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR435,<ENABLE>*CS<CR><LF>
ENABLE: Stationary antenna reference point (Message type 1005)
0: Disable (Default)
1: Enable
[Example]
Send:
$PAIR435*38\r\n
Response:
$PAIR001,435,0*39\r\n ==> Success
$PAIR435,1*25\r\n ==> RTCM3.x output with message type 1005

2.3.91 Packet Type:436 PAIR_RTCM_SET_OUTPUT_EPHEMERIS

This command is to set enable/disable RTCM output with satellite ephemeris.

DataField: $PAIR436,<ENABLE>*CS<CR><LF>			
Name	Unit	Default	Description
ENABLE	--	--	0: Disable 1: Enable

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR436,1*26\r\n ==> set RTCM3.x output with satellite ephemeris
Response:
$PAIR001,436,0*3A\r\n ==> Success

2.3.92 Packet Type:437 PAIR_RTCM_GET_OUTPUT_EPHEMERIS

Query setting of RTCM satellite ephemeris output.

DataField: $PAIR437*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR437,<ENABLE>*CS<CR><LF>
ENABLE:
0: Disable
1: Enable
[Example]
Send:
$PAIR437*3A\r\n
Response:
$PAIR001,437,0*3B\r\n ==> Success
$PAIR437,1*27\r\n ==> RTCM3.x output with satellite ephemeris

2.3.93 Packet Type:470 PAIR_EPO_GET_STATUS

Query the EPO data status stored in the GPS chip

DataField: $PAIR470,<System_ID>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
System_ID	--	--	The GNSS system ID: '0' = GPS '1' = GLONASS '2' = Galileo '3' = BeiDou

Return&Example

[Return]
1. PAIR_ACK for send result.

2. $PAIR470,<System_ID>,<Set>,<FWN>,<FTOW>,<LWN>,<LTOW>,<FCWN>,<FCTOW>,<LCWN>,<LCTOW>*CS<CR><LF>

System_ID: The GNSS system ID.

'0' = GPS

'1' = GLONASS

'2' = Galileo

'3' = BeiDou

Set: Total number sets of EPO data stored in chip

FWN, FTOW: GPS week number & TOW of the first set of EPO data stored in chip respectively (flash)

LWN, LTOW: GPS week number & TOW of the last set of EPO data stored in chip respectively (flash)

FCWN, FCTOW: GPS week number & TOW of the first set of EPO data that are currently used respectively

LCWN, LCTOW: GPS week number & TOW of the last set of EPO data that are currently used respectively

[Example]

Send:

$PAIR470,0*25\r\n

Response:

$PAIR001,470,0*38\r\n ==> Success

$PAIR470,0,1,2098,194400,2098,216000,2098,194400,2098,216000*38\r\n

2.3.94 Packet Type:471 PAIR_EPO_SET_DATA

Send the packet containing EPO data for a single satellite.

DataField: $PAIR471,<System_ID>,<SV_ID>,<W[0]>,...,<W[17]>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
System_ID	--	--	The GNSS system ID: '0' = GPS '1' = GLONASS '2' = Galileo '3' = BeiDou
SV_ID			Satellite PRN number for the EPO data to follow. [Represented in hexadecimal format] GPS Range: 1 ~ 32 GLONASS Range: 1 ~ 24 GALILEO Range: 1 ~ 30 BEIDOU Range: 1 ~ 37 Special 255: BeiDou IONO data. Special 254: Galileo IONO data.

W[0] ~ W[17]			words [LSB first] of one EPO segment data (total 72 bytes).

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR471,<System_ID>,<SV_ID>*CS<CR><LF>
[Example]
Send:
$PAIR471,1,16,56056272,F2BC0244,4F19AE34,F95C534D,FAE67014,4F19AF6B,F96749BD,9F341F2D,6F4EA9F,77DB4710,66ADAC2,9ADF3B01,8CC8B19C,29D2D20C,FC5B2E94,1000001C,11005000,748B45F4*0A\r\n
Response:
$PAIR001,471,0*39\r\n ==> Success

2.3.95 Packet Type:472 PAIR_EPO_ERASE_FLASH_DATA

Erase the EPO data stored in the flash memory

DataField: $PAIR472*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR472*3B\r\n
Response:
$PAIR001,472,0*3A\r\n ==> Success

2.3.96 Packet Type:473 PAIR_EPO_FLASH_AIDING_ENABLE

Enable EPO flash aiding. This feature limits the flash size (Max 80K) to save EPO data.

DataField: $PAIR473,<Enable>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	'1'	Enable EPO flash aiding or not. '0' = Disable this feature. GNSS chip does not limit the flash range for saving EPO data. '1' = Enable this feature. GNSS chip will limits the flash range to 80K for saving EPO data.

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR473,1*27\r\n
Response:
$PAIR001,473,0*3B\r\n ==> Success

2.3.97 Packet Type:490 PAIR_EASY_ENABLE

Enable or disable EASY function

DataField: $PAIR490,<Enable>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	Enable or disable: '0': Disable (Default) '1': Enable

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR490,1*2A\r\n
Response:
$PAIR001,490,0*36\r\n ==> Success

2.3.98 Packet Type:491 PAIR_EASY_GET_STATUS

Query whether EASY is enabled or disabled

DataField: $PAIR491*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]

1. PAIR_ACK for send result.

2. $PAIR490,<Enable>,<Status>*CS<CR><LF>

Enable: Enable or disable

'0': Disable

'1': Enable

Status:

'0': Not finished

'1': finished 1-day extension

'2': finished 2-day extension

'3': finished 3-day extension

[Example]

Send:

$PAIR491*36\r\n

Response:

$PAIR001,491,0*37\r\n ==> Success

$PAIR491,1,0*37\r\n

2.3.99 Packet Type:493 PAIR_EASY_SET_BACKGROUND_ENABLE

To compute EASY data even GNSS subsystem is power off.

DataField: $PAIR493,<Enable>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	'0': Disable '1': Enable

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR493,1*29\r\n
Response:
$PAIR001,493,0*35\r\n

2.3.100 Packet Type:510 PAIR_NVRAM_AUTO_SAVING_ENABLE

Enable/Disable navigation data auto saving from RTC RAM to flash.
SIM65M module will automatically save the data at the first fix and then every 30 minutes.

DataField: $PAIR510,<Enable>*CS<CR><LF>

Name	Unit	Default	Description
Enable	--	--	0: Disable 1: Enable

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR510,1*23\r\n
3 Response:
4 $PAIR001,510,0*3F\r\n

NOTE

This command can only be set in 1Hz.

2.3.101 Packet Type:511 PAIR_NVRAM_SAVE_NAVIGATION_DATA

Save current navigation data from RTC RAM to flash

DataField: $PAIR511*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR511*3F\r\n
Response:
$PAIR001,511,1*3F\r\n ==> Processing
$PAIR001,511,0*3E\r\n ==> Success

NOTE

In multi-Hz, this command can only be set when the GNSS system is powered off,while 1Hz does not have this limitation.

2.3.102 Packet Type:512 PAIR_NVRAM_CLEAR_NAVIGATION_DATA

Clear navigation data in both RTC RAM and flash.
Note: This command is only used for testing.

DataField: $PAIR512*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]

1. PAIR_ACK for send result

[Example]

Send:

$PAIR512*3C\r\n

Response:

$PAIR001,512,0*3D\r\n

2.3.103 Packet Type:513 PAIR_NVRAM_SAVE_SETTING

Save the current configuration from RTC RAM to flash.

DataField: $PAIR513*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]

1. PAIR_ACK for send result

[Example]

Send:

$PAIR513*3D\r\n

Response:

$PAIR001,513,0*3C\r\n

NOTE

You need to send this command every time after modifying any parameters, if the HW not keep RTC power.

Otherwise, the changes will be lost after system reboot and the GNSS module must be reconfigured again.

If HW will keep RTC power, no need to use this command. The change of configuration will keep in the RTC RAM.

In multi-Hz, this command can only be set when the GNSS system is powered off,while 1Hz does not have this limitation.

2.3.104 Packet Type:514 PAIR_NVRAM_RESTORE_DEFAULT_SETTING

Clear the current configuration and restore the default settings.
This function does not support run time restore when GNSS is power on.
Please send **PAIR_GNSS_SUBSYS_POWER_OFF** to power off GNSS before use this command.

DataField: $PAIR514*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR514*3A\r\n
Response:
$PAIR001,514,0*3B\r\n

2.3.105 Packet Type:530 PAIR_EPH_GET_STATUS

Get the EPH status in the next few seconds

DataField: $PAIR530,<Constellation>,<Time_interval>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Constellation	--	--	The GNSS system ID: '0' = GPS '1' = GLONASS '2' = Galileo '3' = BeiDou '4' = QZSS
Time_interval	--	--	The range is between 1 and 7200 seconds (2 hours). The unit is seconds

Example

[Return]

1. PAIR_ACK for send result.
2. $PAIR530,<Constellation>,<L1_SV>,<L5_SV>*CS<CR><LF>
The valid ephemeris SV is in HEX format.
GLONASS only reports <L1_SV>.
Only dual packet reports both <L1_SV> and <L5_SV>.
[Example]
Send:
$PAIR530,1,1800*04\r\n
This command queries the status of GPS ephemeris after 1800 seconds in the future.
Response:
$PAIR001,530,0*3D\r\n ==> Success
$PAIR530,40449464,00800000*3F\r\n
Note the HEX 40449464 means 0100 0000 1000 0100 1001 0100 0110 0100 and the valid L1 SV numbers are 3,6,7,11,13,16,19,24,31, while
the HEX 00800000 means 0000 0000 1000 0000 0000 0000 000 0000 and the valid L5 SV number is 24

2.3.106 Packet Type:531 PAIR_EPH_SET_DATA

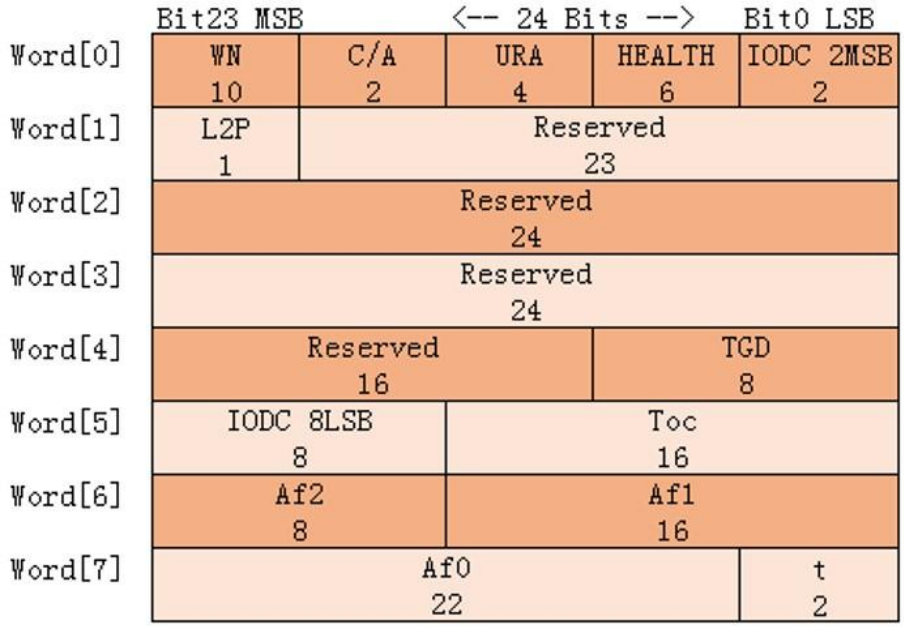
Send ephemeris subframe message to GNSS chip.

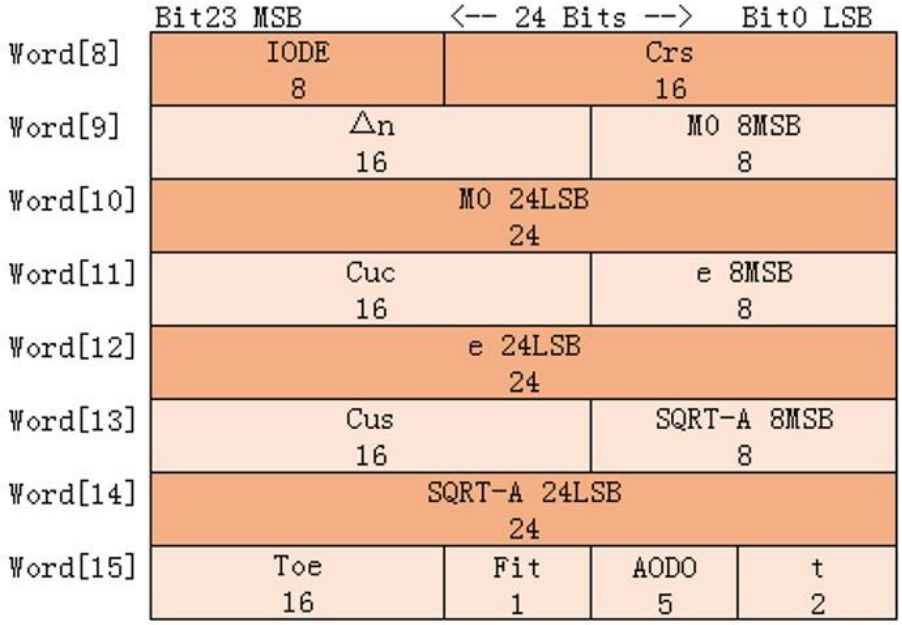
DataField: $PAIR531,<Constellation>,<Signal_ID>,<SV_ID>,<EPH_data>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Constellation	--	--	The GNSS system ID: '0' = GPS '1' = GLONASS '2' = Galileo '3' = BeiDou '4' = QZSS
Signal_ID	--	--	Signal type L1: 0 L5: 1 (including GPS L5, Galileo E5a, BeiDou B2a)
SV_ID			in DEC format GPS: 1-32 GLONASS: 1-24 Galileo: 1-36 BeiDou: 1-63 (GEO: 1-5, 59-63; MEO: 6-58) QZSS: 1-7
EPH_data			The ephemeris data for aiding GPS(L1): W[0],...,W[23] 24 words of the ephemeris subframes data from words 3 to 10 of subframes 1, 2 and 3 of the GPS

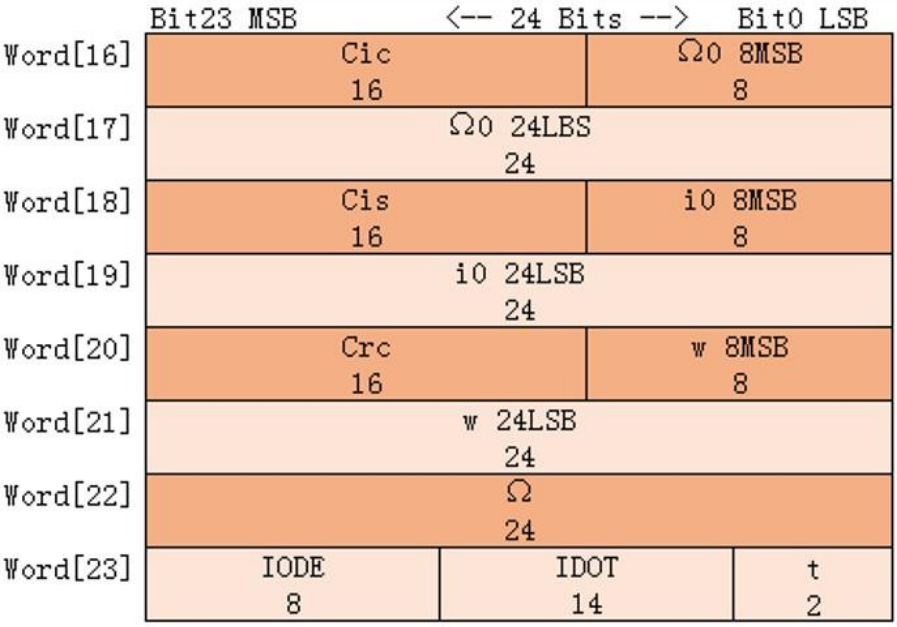
			Navigation Message. Each of the raw 30 bit data words have been logically shifted 6 bits to the right to remove the 6 parity bits leaving the 24 data bits. GPS(L5): W[0],...,W[19] 20 words of the ephemeris subframes data from bits 33 to 276 of message type 10, 11 and 30 of the GPS Navigation Message. Every item contains 32 bits, every eight items make up a message type. The last four items are the clock data. GLONASS(L1): W[0],...,W[15] First 15 words of the ephemeris subframes data from strings 1 to 5 of the GLO Navigation Message. Every item contains 32 bits, every three items make up a string. The first item of a string contains bit1-32. The second item of a string contains bit33-64. The third item of a string contains bit65-72, the last 8 bits of item is valid. The Last word, W[16], indicates the frequency channel, which range from 1 to 14 in HEX format. Galileo(L1): W[0],...,W[15],(W[16],...,W[18]) 19 words in total including 16 words of the ephemeris subframes data from word types 1 to 4 of the Galileo Navigation Message and 3 words from word type 5 with BGD, health, data valid status,and GST. Note that the user only needs to input W[0],...,W[15] to set EPH, while W[0],...,W[18] will be shown when getting EPH. The word type (6-bit) and IODnav (10-bit) have been removed and shifted to the right. Every item contains 32 bits, every four items make up a word type except W[16],...,W[18]. The first item of a word type contains bit81-112. The second item of a word type contains bit49-80. The third item of a word type contains bit17-48. The fourth item of a word type contains bit1-16, the last 16 bits of item is valid. Galileo(E5a): W[0],...,W[31] 32 words of the ephemeris subframes data from word types 1 to 4 of the Galileo E5a Navigation Message. Every item contains 32 bits, every eight items make up a word type. The first item of a word type contains bit217-248. The second item of a word type contains bit185-216,

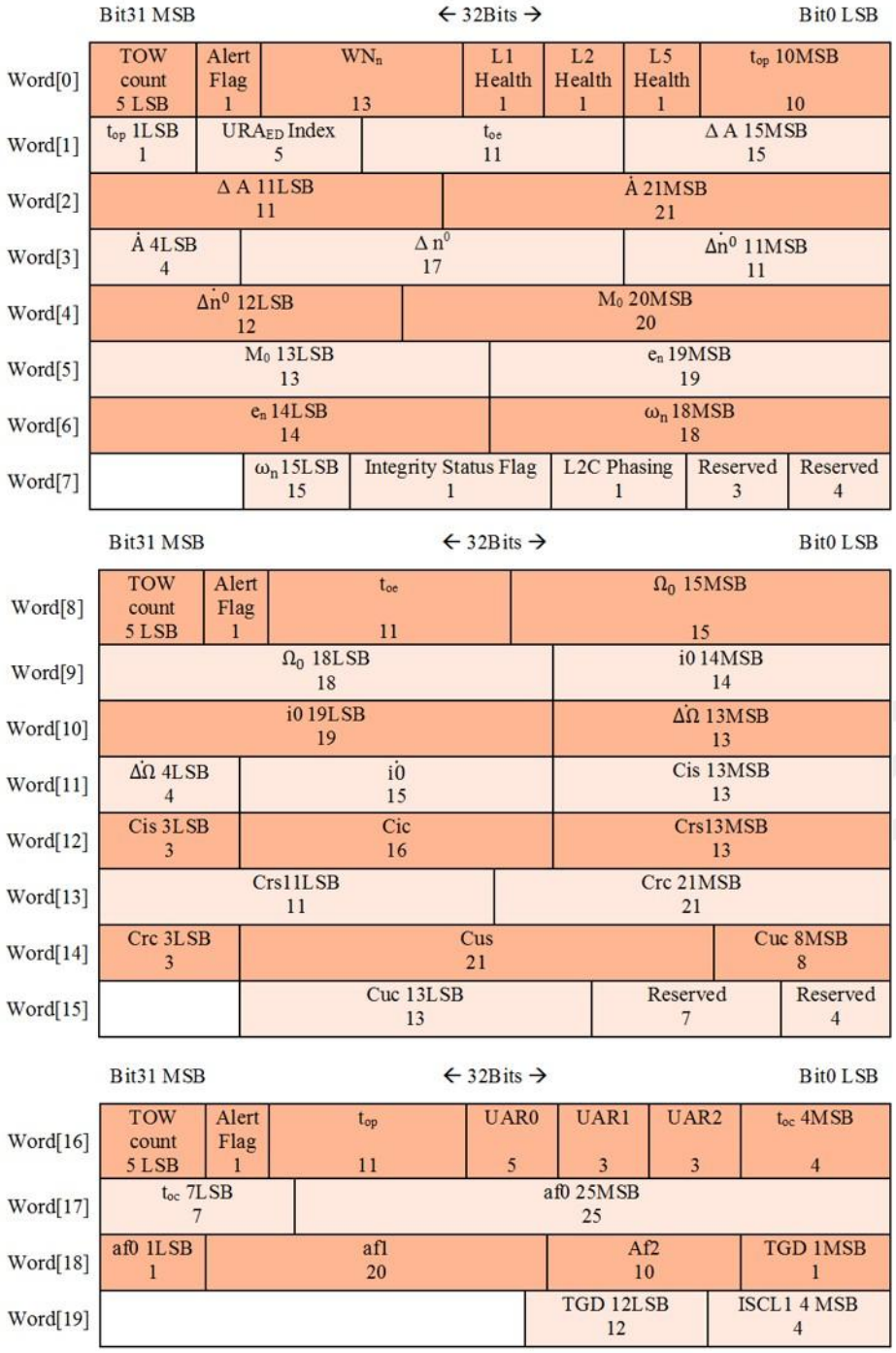
			and so on. The eighth item of a word type contains bit1-24, the last 24 bits of item is valid. BeiDou GEO(L1): W[0],...,W[29] 30 words of the ephemeris subframes data from subframe 1 page 1 to 10 of the BeiDou GEO Navigation Message. The 8 most significant parity-bits have been removed and shifted to the right. Every item contains 32 bits, every three items make up a page. The first item of a page contains bit41-72. The second item of a page contains bit9-40. The third item of a page contains bit1-8, the last 8 bits of item is valid. BeiDou MEO(L1): W[0],...,W[20] 21 words of the ephemeris subframes data from subframes 1 to 3 of the BeiDou MEO Navigation Message. The 8 most significant parity-bits have been removed and shifted to the right. Every item contains 32 bits, every seven items make up a subframe. The first item of a subframe contains bit193-224. The second item of a subframe contains bit161-192, and so on. The seventh item of a subframe contains bit1-32. BeiDou MEO(B2a): W[0],...,W[20] 21 words of the ephemeris subframes data from the BeiDou B2a Navigation Message. The CRC (24-bit) has been removed and shifted to the right. Every item contains 32 bits, every nine items make up a message type. The first item of a message type contains bit233-264. The second item of a message type contains bit201-232, and so on. The ninth item of a message type contains bit1-8, the last 8 bits of item is valid. QZSS(L1): W[0],...,W[23] 24 words of the ephemeris subframes data from words 3 to 10 of subframes 1, 2 and 3 of the QZSS Navigation Message. Each of the raw 30 bit data words have been logically shifted 6 bits to the right to remove the 6 parity bits leaving the 24 data bits.

			QZSS(L5): W[0],...,W[19] 20 words of the ephemeris subframes data from bits 33 to 276 of message type 10, 11 and 30 of the QZSS Navigation Message. Every item contains 32 bits, every eight items make up a message type. The last four items are the clock data.


Bit23 MSB
<-- 24 Bits -->
Bit0 LSB
Word[0]
WN 10
C/A 2
URA 4
HEALTH 6
IODC 2MSB 2
Word[1]
L2P 1
Reserved 23
Word[2]
Reserved 24
Word[3]
Reserved 24
Word[4]
Reserved 16
TGD 8
Word[5]
IODC 8LSB 8
Toc 16
Word[6]
Af2 8
Af1 16
Word[7]
Af0 22
t 2


Bit23 MSB
<-- 24 Bits -->
Bit0 LSB
Word[8]
IODE 8
Crs 16
Word[9]
Δn 16
M0 8MSB 8
Word[10]
M0 24LSB 24
Word[11]
Cuc 16
e 8MSB 8
Word[12]
e 24LSB 24
Word[13]
Cus 16
SQRT-A 8MSB 8
Word[14]
SQRT-A 24LSB 24
Word[15]
Toe 16
Fit 1
AODO 5
t 2

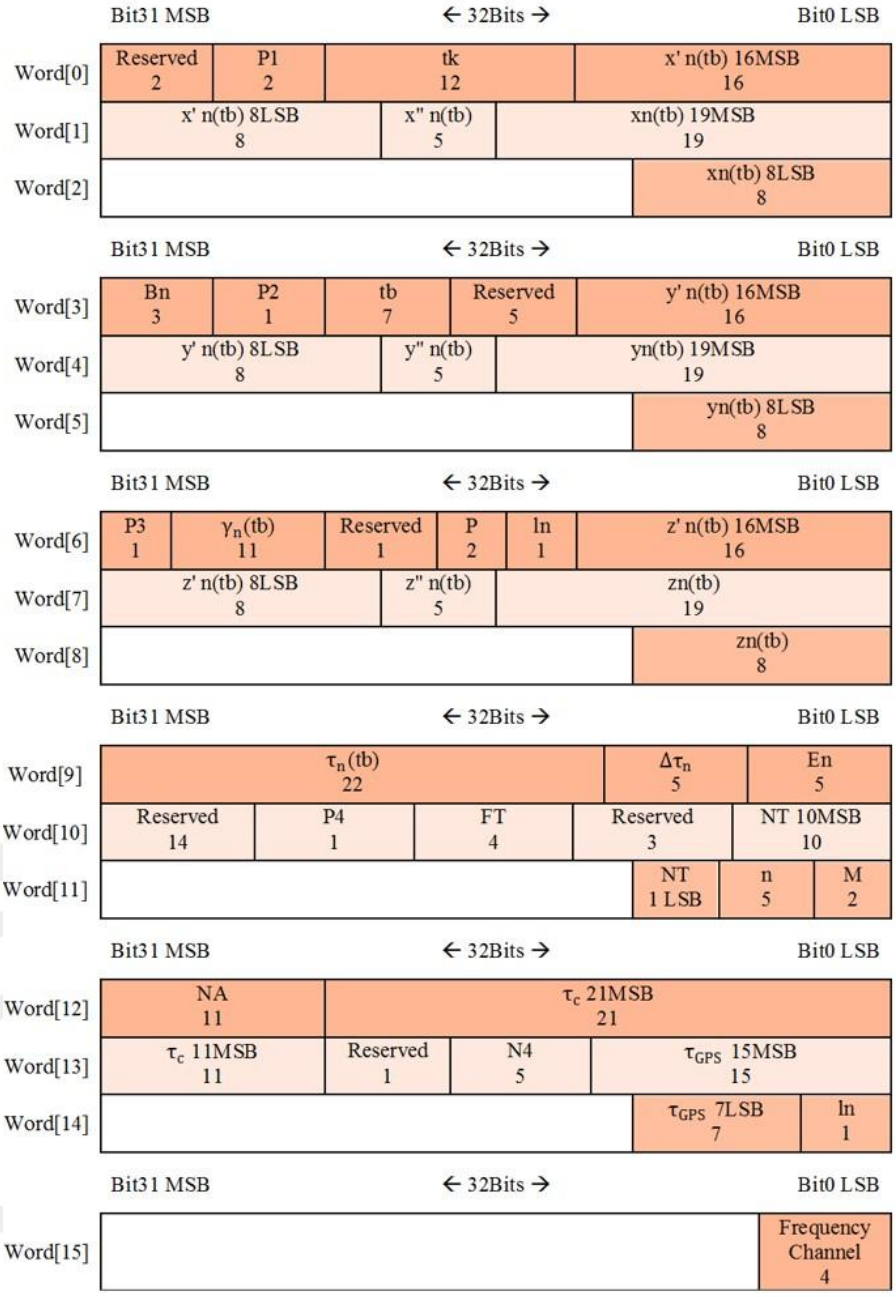

Bit23 MSB
<-- 24 Bits -->
Bit0 LSB
Word[16]
Cic 16
Ω0 8MSB 8
Word[17]
Ω0 24LBS 24
Word[18]
Cis 16
i0 8MSB 8
Word[19]
i0 24LSB 24
Word[20]
Crc 16
w 8MSB 8
Word[21]
w 24LSB 24
Word[22]
Ω 24
Word[23]
IODE 8
IDOT 14
t 2

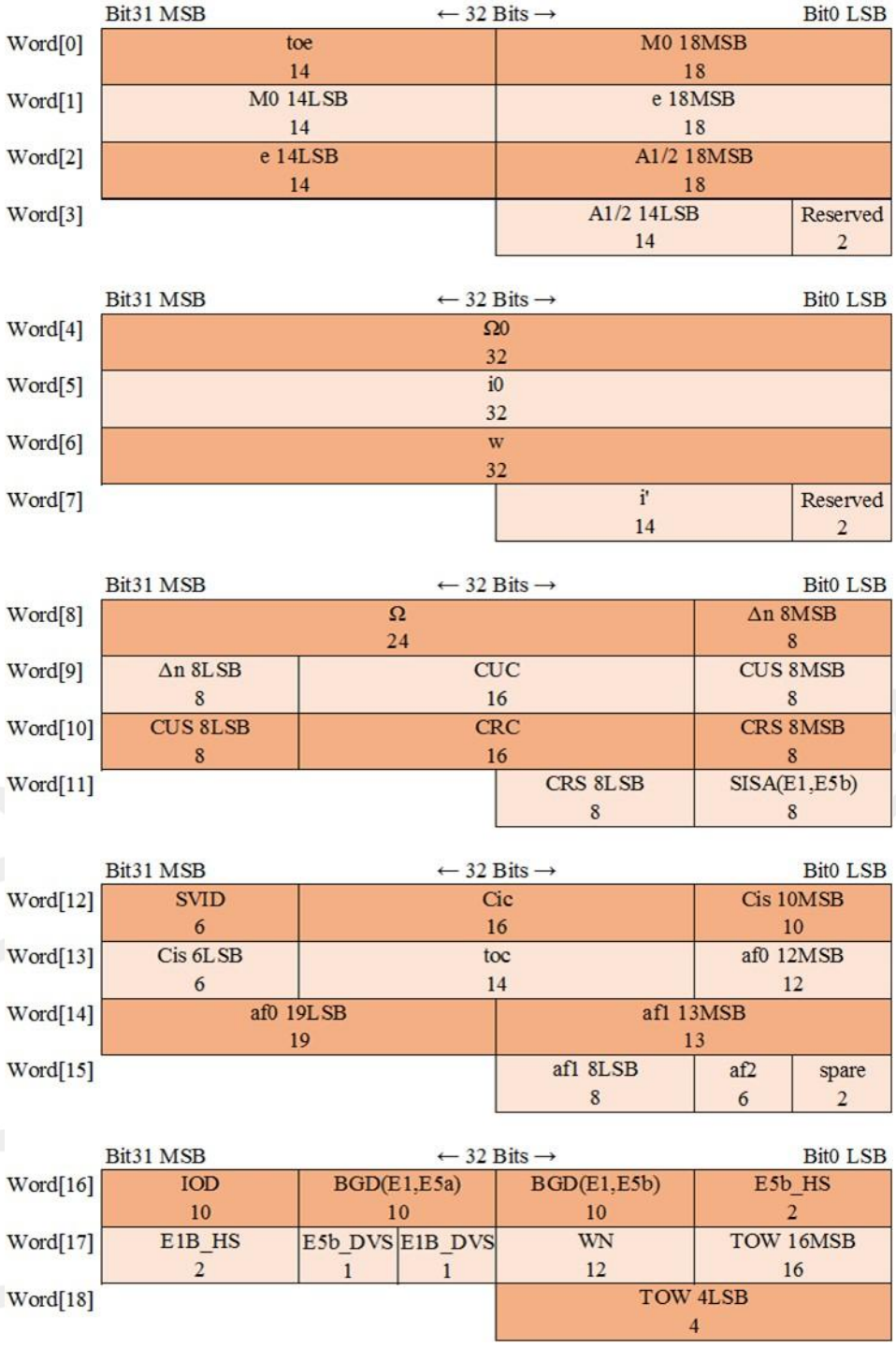


	Bit31 MSB ← 32Bits → Bit0 LSB						
Word[0]	TOW count 5 LSB	Alert Flag 1	WN_n 13	L1 Health 1	L2 Health 1	L5 Health 1	t_{op} 10MSB 10
Word[1]	t_{op} 1LSB 1	URA_{ED} Index 5	t_{oe} 11	Δ A 15MSB 15			
Word[2]	Δ A 11LSB 11	Ȧ 21MSB 21					
Word[3]	Ȧ 4LSB 4	Δn^0 17	$\Delta\dot{n}^0$ 11MSB 11				
Word[4]	$\Delta\dot{n}^0$ 12LSB 12	M_0 20MSB 20					
Word[5]	M_0 13LSB 13	e_n 19MSB 19					
Word[6]	e_n 14LSB 14	ω_n 18MSB 18					
Word[7]		ω_n 15LSB 15	Integrity Status Flag 1	L2C Phasing 1	Reserved 3	Reserved 4	

	Bit31 MSB ← 32Bits → Bit0 LSB						
Word[8]	TOW count 5 LSB	Alert Flag 1	t_{oe} 11	Ω_0 15MSB 15			
Word[9]	Ω_0 18LSB 18	i0 14MSB 14					
Word[10]	i0 19LSB 19	$\Delta\dot{\Omega}$ 13MSB 13					
Word[11]	$\Delta\dot{\Omega}$ 4LSB 4	$\dot{i0}$ 15	Cis 13MSB 13				
Word[12]	Cis 3LSB 3	Cic 16	Crs13MSB 13				
Word[13]	Crs11LSB 11	Crc 21MSB 21					
Word[14]	Crc 3LSB 3	Cus 21	Cuc 8MSB 8				
Word[15]		Cuc 13LSB 13	Reserved 7	Reserved 4			

	Bit31 MSB ← 32Bits → Bit0 LSB						
Word[16]	TOW count 5 LSB	Alert Flag 1	t_{op} 11	UAR0 5	UAR1 3	UAR2 3	t_{oc} 4MSB 4
Word[17]	t_{oc} 7LSB 7	af0 25MSB 25					
Word[18]	af0 1LSB 1	af1 20	Af2 10	TGD 1MSB 1			
Word[19]		TGD 12LSB 12	ISCL1 4 MSB 4				


Bit31 MSB
← 32Bits →
Bit0 LSB
Word[0]
Reserved 2
P1 2
tk 12
x' n(tb) 16MSB 16
Word[1]
x' n(tb) 8LSB 8
x'' n(tb) 5
xn(tb) 19MSB 19
Word[2]
xn(tb) 8LSB 8
Bit31 MSB
← 32Bits →
Bit0 LSB
Word[3]
Bn 3
P2 1
tb 7
Reserved 5
y' n(tb) 16MSB 16
Word[4]
y' n(tb) 8LSB 8
y'' n(tb) 5
yn(tb) 19MSB 19
Word[5]
yn(tb) 8LSB 8
Bit31 MSB
← 32Bits →
Bit0 LSB
Word[6]
P3 1
γn(tb) 11
Reserved 1
P 2
ln 1
z' n(tb) 16MSB 16
Word[7]
z' n(tb) 8LSB 8
z'' n(tb) 5
zn(tb) 19
Word[8]
zn(tb) 8
Bit31 MSB
← 32Bits →
Bit0 LSB
Word[9]
τn(tb) 22
Δτn 5
En 5
Word[10]
Reserved 14
P4 1
FT 4
Reserved 3
NT 10MSB 10
Word[11]
NT 1 LSB
n 5
M 2
Bit31 MSB
← 32Bits →
Bit0 LSB
Word[12]
NA 11
τc 21MSB 21
Word[13]
τc 11MSB 11
Reserved 1
N4 5
τGPS 15MSB 15
Word[14]
τGPS 7LSB 7
ln 1
Bit31 MSB
← 32Bits →
Bit0 LSB
Word[15]
Frequency Channel 4


Bit31 MSB
← 32 Bits →
Bit0 LSB
Word[0]
toe 14
M0 18MSB 18
Word[1]
M0 14LSB 14
e 18MSB 18
Word[2]
e 14LSB 14
A1/2 18MSB 18
Word[3]
A1/2 14LSB 14
Reserved 2
Word[4]
Ω0 32
Word[5]
i0 32
Word[6]
w 32
Word[7]
i' 14
Reserved 2
Word[8]
Ω 24
Δn 8MSB 8
Word[9]
Δn 8LSB 8
CUC 16
CUS 8MSB 8
Word[10]
CUS 8LSB 8
CRC 16
CRS 8MSB 8
Word[11]
CRS 8LSB 8
SISA(E1,E5b) 8
Word[12]
SVID 6
Cic 16
Cis 10MSB 10
Word[13]
Cis 6LSB 6
toc 14
af0 12MSB 12
Word[14]
af0 19LSB 19
af1 13MSB 13
Word[15]
af1 8LSB 8
af2 6
spare 2
Word[16]
IOD 10
BGD(E1,E5a) 10
BGD(E1,E5b) 10
E5b_HS 2
Word[17]
E1B_HS 2
E5b_DVS 1
E1B_DVS 1
WN 12
TOW 16MSB 16
Word[18]
TOW 4LSB 4

	Bit31 MSB ← 32Bits → Bit0 LSB								
Word[0]	Type = 1 6	SVID 6	IOD_{nav} 10	toc 10MSB 10					
Word[1]	toc 4LSB 4	af0 29MSB 29							
Word[2]	af0 3LSB 3	af1 21		af2 6	SISA 2MSB 2				
Word[3]	SISA 6LSB 6	ai0 11		ai1 11	ai2 4MSB 4				
Word[4]	ai2 10LSB	Region 1 1	Region 2 1	Region 3 1	Region 4 1	Region 5 1	BGD 10	$E5a_{HS}$ 2	WN 5MSB
Word[5]	WN 7LSB 7	TOW 20					$E5a_{DVS}$ 1	Spare 4MSB 4	
Word[6]	Spare 22LSB 22						CRC 10MSB 10		
Word[7]		CRC 14LSB 14					Tail 6	Reserved 4	

	Bit31 MSB ← 32Bits → Bit0 LSB								
Word[8]	Type = 2 6	IOD_{nav} 10	M0 16MSB 16						
Word[9]	M0 16LSB 16	$\dot{\Omega}$ 16MSB 16							
Word[10]	$\dot{\Omega}$ 8LSB 8	e 24MSB 24							
Word[11]	e 8LSB 8	$\sqrt{A}$ 24MSB 24							
Word[12]	$\sqrt{A}$ 8LSB 8	Ω_0 24MSB 24							
Word[13]	Ω_0 8LSB 8	i 14					WN 10MSB 10		
Word[14]	WN 2LSB 2	TOW 20					CRC 10MSB 10		
Word[15]		CRC 14LSB 14					Tail 6	Reserved 4	

	Bit31 MSB ← 32Bits → Bit0 LSB			
Word[16]	Type = 3 6	IOD_{nav} 10	I0 16MSB 16	
Word[17]	i0 16LSB 16		ω16MSB 16	
Word[18]	ω16LSB 16		Δn 16	
Word[19]	Cuc 16		Cus 16	
Word[20]	Crc 16		Crs 16	
Word[21]	t_{oe} 14	WN 12	TOW 6MSB 6	
Word[22]	TOW 14LSB 14	Spare 8	CRC 10MSB 10	
Word[23]		CRC 14LSB 14	Tail 6	Reserved 4

	Bit31 MSB ← 32Bits → Bit0 LSB			
Word[24]	Type = 4 6	IOD_{nav} 10	Cic 16	
Word[25]	Cis 16		A0 16MSB 16	
Word[26]	A0 16LSB 16		A1 16MSB 16	
Word[27]	A1 8LSB 8	Δt_{LS} 8	t_{0t} 8	WN_{0t} 8
Word[28]	WN_{LSF} 8	DN 3	Δt_{LSF} 8	t_{0G} 8	A_{0t} 5MSB 5
Word[29]	A_{0t} 11LSB 11	A_{1G} 12	WN_{0G} 6	TOW 3MSB 3
Word[30]	TOW 17LSB 17	Spare 5	CRC 10MSB 10	
Word[31]		CRC 14LSB 14	Tail 6	Reserved 4

	Bit31 MSB <-- 32 Bits --> Bit0 LSB				
Word[0]	SatH1 1	AODC 5	URAI 4	WN 13	Toc 9MSB 9
Word[1]	Toc 8LSB 8	TGD1 10	TGD2 10		Reserved 4
Word[2]				Reserved 8	

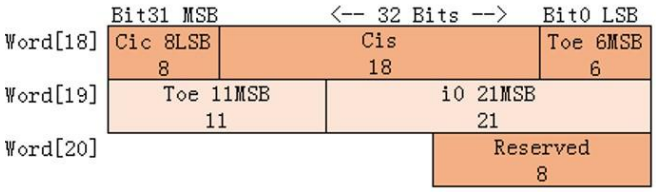
	Bit31 MSB <-- 32 Bits --> Bit0 LSB			
Word[9]	a1 18		a2 11	AODE 3MSB 3
Word[10]	AODE 2LSB 2	n 16		Cuc 14MSB 14
Word[11]			Reserved 8	

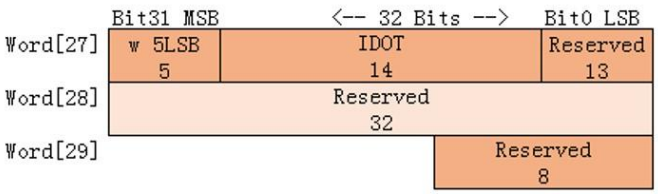
	Bit31 MSB <-- 32 Bits --> Bit0 LSB			
Word[3]	Alpha0 8	Alpha1 8	Alpha2 8	Alpha3 8
Word[4]	Beta0 8	Beta1 8	Beta2 8	Beta3 8
Word[5]			Reserved 8	

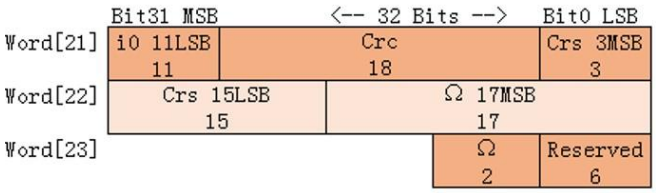
	Bit31 MSB <-- 32 Bits --> Bit0 LSB			
Word[12]	Cuc 4LSB 4	M0 28MSB 28		
Word[13]	M0 4LSB 4	Cus 18		e 10MSB 10
Word[14]			Reserved 8	

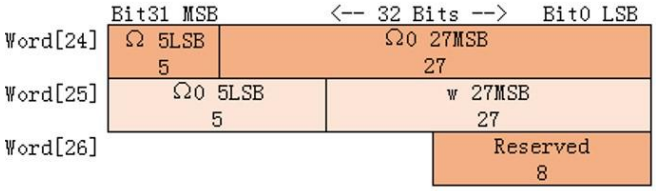
	Bit31 MSB <-- 32 Bits --> Bit0 LSB			
Word[6]	Reserved 32			
Word[7]	Reserved 6	a0 24		a1 2MSB 2
Word[8]			a1 2LSB 2	Reserved 6

	Bit31 MSB <-- 32 Bits --> Bit0 LSB			
Word[15]	e 22LSB 22		SQRT-A 10	
Word[16]	SQRT-A 22			Cic 10MSB 10
Word[17]			Reserved 8	





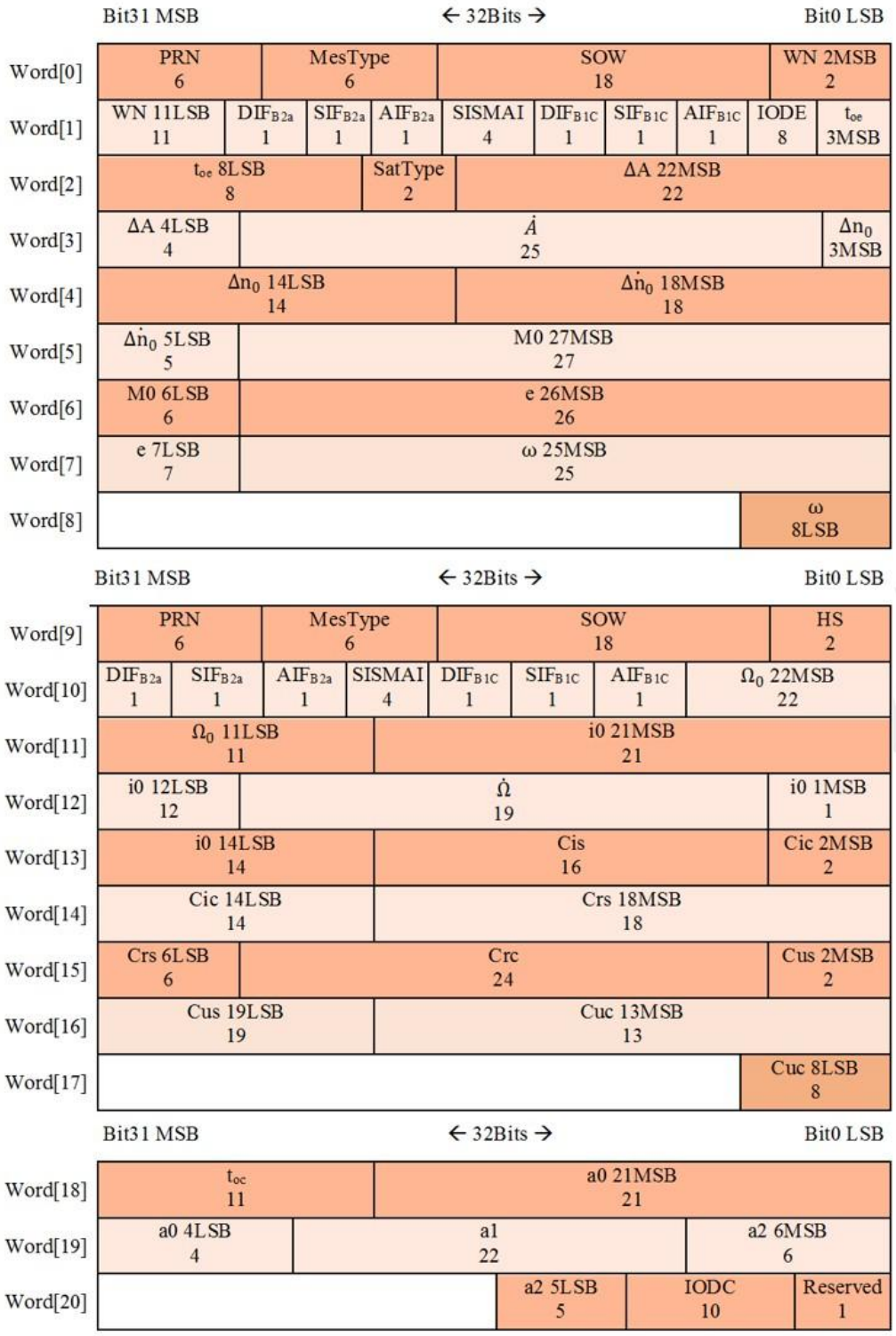




	Bit31 MSB		<-- 32 Bits -->			Bit0 LSB
Word[0]	Pre 11		Reserved 4	FraID 3	SOW 14MSB 14	
Word[1]	SOW 6LSB 6	SatH1 1	AODC 5	URAI 4	WN 13	toc 3MSB 3
Word[2]	toc 14LSB 14		TGD1 10		TGD2 8MSB 8	
Word[3]	TGD2 2LSB 2	Alpha0 8	Alpha1 8	Alpha2 8		Alpha3 6MSB 6
Word[4]	Alpha3 2LSB 2	Beta0 8	Beta1 8	Beta2 8		Beta3 6MSB 6
Word[5]	Beta3 2LSB 2	a2 11		a0 19MSB 19		
Word[6]	a0 5LSB 5	a1 22				AODE 5

	Bit31 MSB		<-- 32 Bits -->		Bit0 LSB
Word[7]	Pre 11	Reserved 4	FraID 3	SOW 14MSB 14	
Word[8]	SOW 6LSB 6	Detan 16			Cuc 10MSB 10
Word[9]	Cuc 8LSB 8	M0 24MSB 24			
Word[10]	M0 8LSB 8	e 24MSB 24			
Word[11]	e 8LSB 8	Cus 18			Crc 6MSB 6
Word[12]	Crc 12LSB 12	Crs 18			SQRT-A 2MSB 2
Word[13]	SQRT-A 30LSB 30				Toe 2MSB 2

	Bit31 MSB		<-- 32 Bits -->		Bit0 LSB
Word[14]	Pre 11	Reserved 4	FraID 3	SOW 14MSB 14	
Word[15]	SOW 6LSB 6	Toe 15LSB 15			i0 11MSB 11
Word[16]	i0 21LSB 21				Cic 11MSB 11
Word[17]	Cic 7LSB 7	Ω 24			Cis 1MSB 1
Word[18]	Cis 17LSB 17	IDOT 14			Ω0 1MSB 1
Word[19]	Ω0 31LSB 31				ω 1MSB 1
Word[20]	ω 31LSB 31				Reserved 1



Example

[Return]

1. PAIR_ACK for send result.

2. $PAIR531,<Status>*CS<CR><LF>

Status: 1 ==> success; 0 ==>fail.

[Example]

Send:

$PAIR531,0,0,1,025000,2B072D,F3002F,4BBD3E,06510C,488598,00FFAB,FA8C41,48F752,28BC4B,654D79,F88804,937C14,1969A1,0D4B91,85987C,FFA27C,508DD6,000F27,C7053B,133E1D,319E79,FFAC83,481070*59\r\n

Response:

$PAIR001,531,0*3C\r\n ==> Success

$PAIR531,1*20\r\n

2.3.107 Packet Type:532 PAIR_EPH_GET_DATA

Get a single ephemeris subframe message.

DataField: $PAIR532,<Constellation>,<Signal_ID>,<SV_ID>*CS<CR><LF>

Name	Unit	Default	Description
Constellation	--	--	The GNSS system ID: '0' = GPS '1' = GLONASS '2' = Galileo '3' = BeiDou '4' = QZSS
Signal_ID	--	--	Signal type L1: 0 L5: 1 (including GPS L5, Galileo E5a, BeiDou B2a)
SV_ID			in DEC format GPS: 1-32 GLONASS: 1-24 Galileo: 1-36 BeiDou: 1-63 (GEO: 1-5, 59-63; MEO: 6-58) QZSS: 1-7

Example

[Return]

1. PAIR_ACK for send result.

2. $PAIR532,<Constellation>,<Signal_ID>,<SV_ID>,<EPH_data>*CS<CR><LF>

EPH_data: refer to the format of PAIR531

[Example]

Send:

$PAIR532,0,0,1*23\r\n

Response:

$PAIR001,532,0*3F\r\n ==> Success

$PAIR532,0,0,01,025000,2B072D,F3002F,4BBD3E,06510C,488598,00FFAB,FA8C41,48F752,

28BC4B,654D79,F88804,937C14,1969A1,0D4B91,85987C,FFA27C,508DD6,000F27,C7053B,133E1D,319E79,FFAC83,481070*6A\r\n

2.3.108 Packet Type:533 PAIR_EPH_CLEAR

Clear the ephemeris data in the critical memory area

DataField: $PAIR533*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Example

```
[Return]
    1. PAIR_ACK for send result.
[Example]
Send:
      $PAIR533*3F\r\n
Response:
      $PAIR001,533,0*3E\r\n ==> Success
```

2.3.109 Packet Type:534 PAIR_EPH_NOTIFY_ENABLE

Enable/Disable notification for newly updated EPH.

DataField: $PAIR534,<Enable>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	0: Disable 1: Enable

Example

```
[Return]
    1. PAIR_ACK for send result.
[Example]
Send:
```

$PAIR534,1*25\r\n
Response:
$PAIR001,534,0*39\r\n ==> Success

2.3.110 Packet Type:535 PAIR_EPH_NOTIFY

The notification of newly updated EPH.

DataField: $PAIR535,<Constellation>,<Signal_ID>,<SV>*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Example

[Return]
$PAIR535,<Constellation>,<Signal_ID>,<SV>*CS<CR><LF>
Constellation: The GNSS system ID.
'0' = GPS
'1' = GLONASS
'2' = Galileo
'3' = BeiDou
'4' = QZSS
Signal_ID: Signal type
L1: 0
L5: 1 (including GPS L5, Galileo E5a, BeiDou B2a)
SV: A bitmap to show the updated EPH of specific SVs (in HEX format)
There are 64 bits in total
[Example]
Response:
$PAIR535,0,0,0000000010080000*1C<CR><LF>
The EPH of GPS L1 PRN20, PRN29 is updated.

NOTE

This command is automatically sent by the GNSS system when PAIR_EPH_NOTIFY_ENABLE is enabled. Please do not actively send it to the GNSS system.

2.3.111 Packet Type:550 PAIR_ALM_GET_STATUS

Get the ALM status in the next few days

DataField: $PAIR550,<Constellation>,<Time_interval>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Constellation	--	--	The GNSS system ID: '0' = GPS '1' = GLONASS '2' = Galileo '3' = BeiDou '4' = QZSS
Time_interval	--	--	Time_interval: The range is between 1 and 91 days. The unit is day

Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR550,<Constellation>,<L1_SV>,<Midi_SV>*CS<CR><LF>
The valid almanac SV is in HEX format.
GLONASS only reports <L1_SV>.
Only dual packet reports both <L1_SV> and <Midi_SV>.
[Example]
Send:
$PAIR550,0,30*09\r\n
This command queries the status of the GPS almanac after 30 days in the future.
Response:
$PAIR001,550,0*3B\r\n ==> Success
$PAIR550,0,FEC0BFFF,00000FFF*24\r\n
Note the HEX FEC0BFFF means 1111 1110 1100 0000 1011 1111 1111 1111 and the valid L1 SV numbers are 1,2,3,4,5,6,7,8,9,10,11,12,13,14,16,23,24,26,27,28,29,30,31,32.
The HEX 00000FFF means 0000 0000 0000 0000 0000 1111 1111 1111 and the valid Midi almanac SV numbers are 1,2,3,4,5,6,7,8,9,10,11,12.

2.3.112 Packet Type:551 PAIR_ALM_SET_DATA

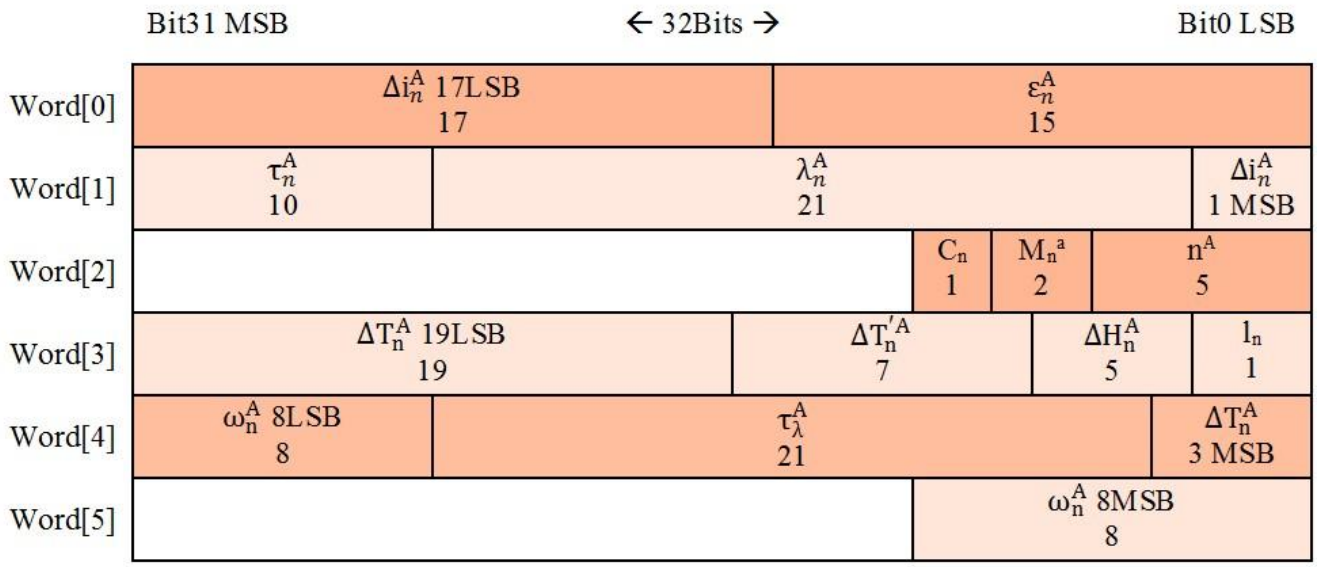
Send the almanac subframe message to GNSS chip.

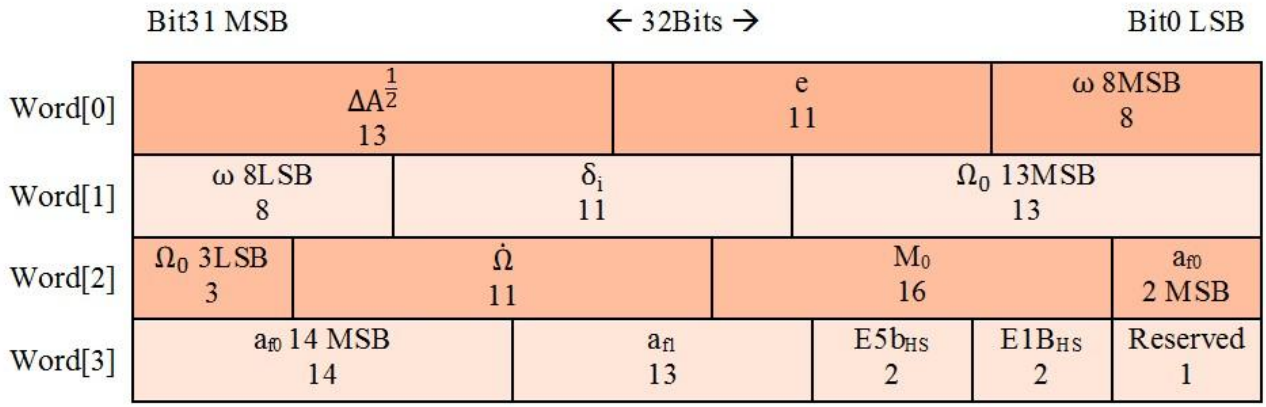
DataField: $PAIR551,<Constellation>,<Signal_ID>,<SV_ID>,<WeekNo>,<ALM_data>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Constellation	--	--	The GNSS system ID: '0' = GPS '1' = GLONASS '2' = Galileo '3' = BeiDou '4' = QZSS
Signal_ID	--	--	Signal type L1: 0 Midi: 1 (including GPS L5, Galileo E5a, BeiDou B2a)
SV_ID			in DEC format GPS: 1-32 GLONASS: 1-24 Galileo: 1-36 BeiDou: 1-63 (GEO: 1-5, 59-63; MEO: 6-58) QZSS: 1-7
WeekNo			in HEX format Almanac reference week number
ALM_data			The almanac data for aiding GPS(L1): W[0],...,W[7] 8 words of the almanac subframes data from pages 1-24 of subframe 5, as well as pages 2-5 and 7-9 of subframe 4 in the GPS Navigation Message. Each of the raw 30 bit data words have been logically shifted 6 bits to the right to remove the 6 parity bits leaving the 24 data bits. GPS(Midi): W[0],...,W[5] 6 words of the almanac subframes data from messtype 37 in the GPS Navigation Message. Every word contains 32 bits, the 8 bits of last word is valid. GLONASS(L1): W[0],...,W[5] 6 words of the almanac subframes data from strings 6 to 15 of the GLO Navigation Message. Every item contains 32 bits, every three items make up a string. The first item of a string contains bit1-32. The second item of a string contains bit33-64. The third item of a string contains bit65-72, the last 8 bits of item is valid. Galileo(L1): W[0],...,W[3] 4 words of the almanac subframes data from word type 7-10 in the Galileo Navigation Message.

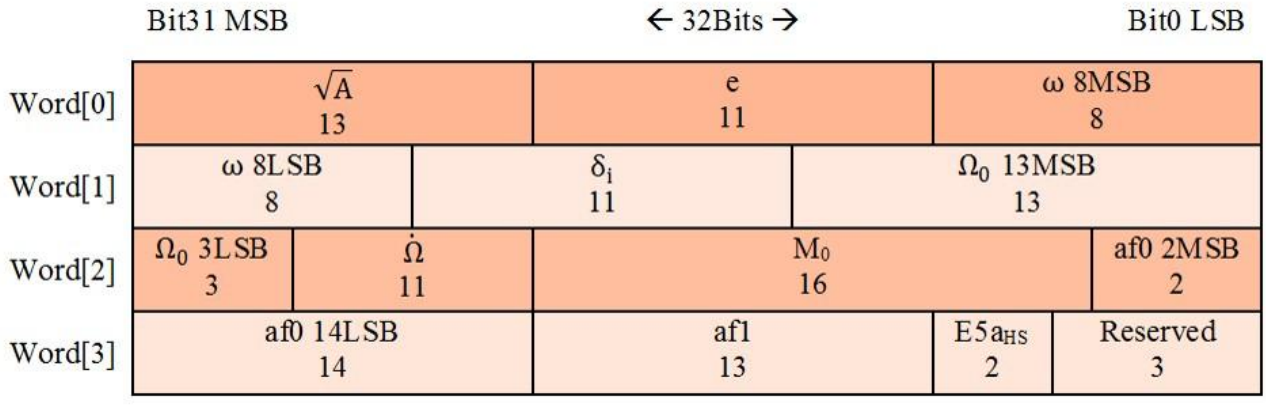
			Every word contains 32 bits, every 4 words make up almanac data of one satellite. Galileo(E5a): W[0],...,W[3] 4 words of the almanac subframes data from word type 5 & 6 in the Galileo Navigation Message. Every word contains 32 bits, every 4 words make up almanac data of one satellite. BeiDou (L1): W[0],...,W[6] 6 words of the almanac subframes data from page 37-60, 95-100 in subframe 5(GEO), page 1-24 in subframe 4 and page 1-6 in subframe 5(MEO). The parity bits have been removed. BeiDou MEO(B2a Midi): W[0],...,W[8] 9 words of the almanac subframes data from message type 40 of the BeiDou Navigation Message. Every word contains 32 bits, the 8 bits of last word is valid. QZSS(L1): W[0],...,W[7] 8 words of the almanac subframes data from pages 1-24 of subframe 5, as well as pages 2-5 and 7-9 of subframe 4 in the GPS Navigation Message. Each of the raw 30 bit data words have been logically shifted 6 bits to the right to remove the 6 parity bits leaving the 24 data bits. QZSS(Midi): W[0],...,W[5] 6 words of the almanac subframes data from messtype 37 in the GPS Navigation Message. Every word contains 32 bits, the 8 bits of last word is valid.

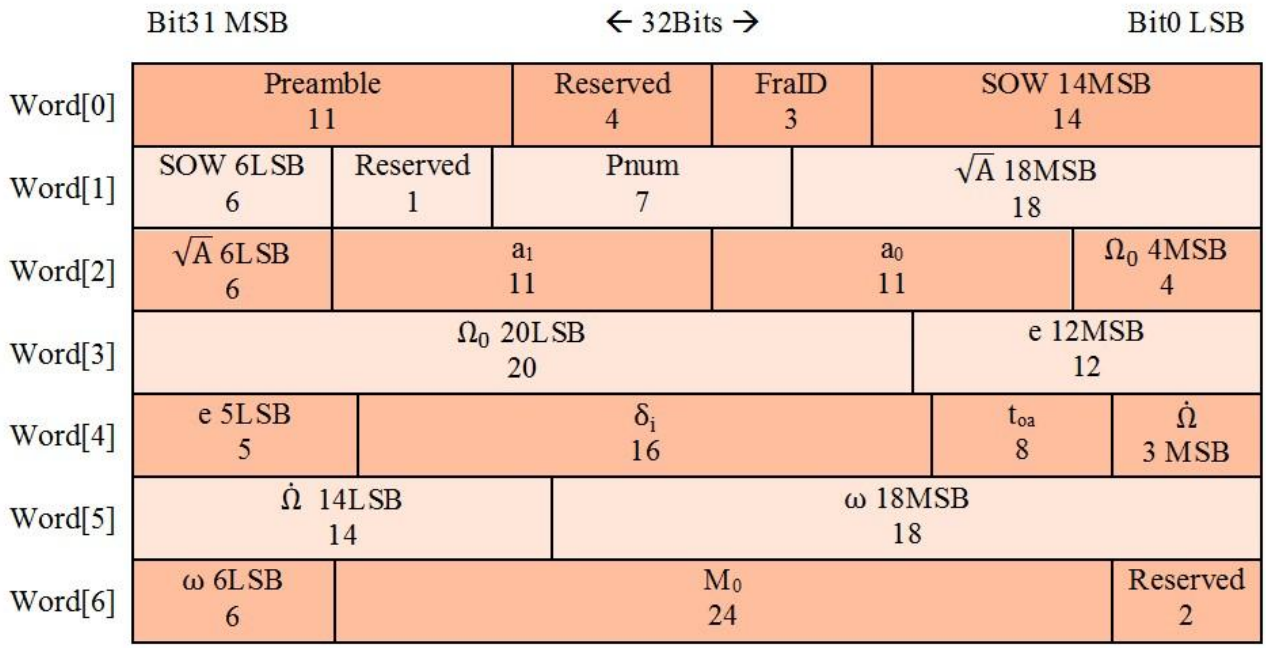
	Bit23 MSB ← 24Bits → Bit0 LSB						
Word[0]	Data ID 2	SV ID 6	e 16				
Word[1]	Toa 8		δ_i 16				
Word[2]	$\dot{\Omega}$ 16				SV Health 8		
Word[3]	$\sqrt{A}$ 24						
Word[4]	Ω_0 24						
Word[5]	ω 24						
Word[6]	M_0 24						
Word[7]	af_0 8MSB 8		af_1 11			af_0 3LSB 3	t 2

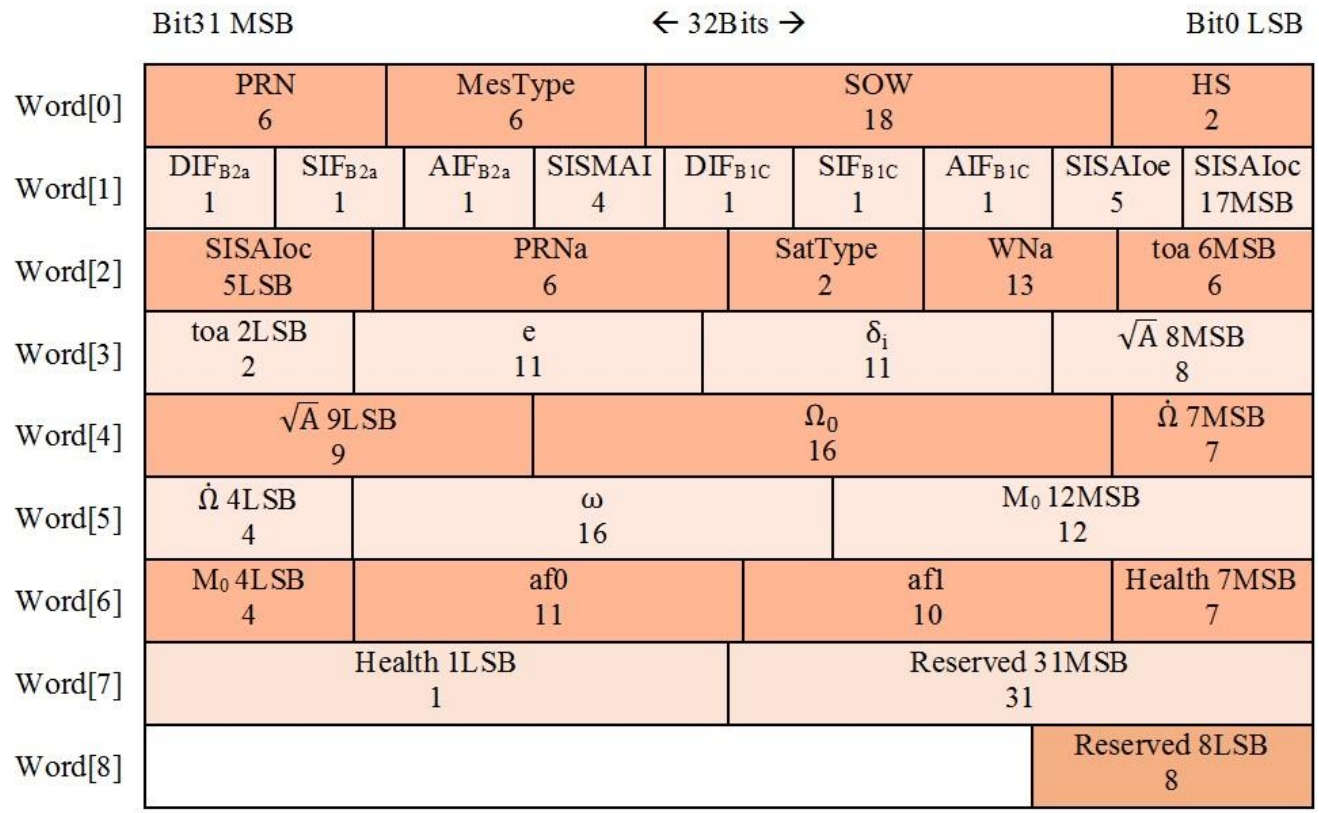
	Bit23 MSB ← 24Bits → Bit0 LSB						
Word[0]	Reserved 4	toa 8	PRN 6	L1 Health 1	L2 Health 1	L5 Health 1	e 3MSB 3
Word[1]	e 8LSB 8		δ_i 11			$\dot{\Omega}$ 5MSB 5	
Word[2]	$\dot{\Omega}$ 6LSB 6	$\sqrt{A}$ 17				Ω_0 1MSB 1	
Word[3]	Ω_0 15LSB 15					ω 9MSB 9	
Word[4]	ω 7LSB 7	M_0 16				af0 1MSB 1	
Word[5]	af0 10LSB 10		af1 10			Reserved 4	


Bit31 MSB
← 32Bits →
Bit0 LSB
Word[0]
Δi_n^A 17LSB 17
ε_n^A 15
Word[1]
τ_n^A 10
λ_n^A 21
Δi_n^A 1 MSB
Word[2]
C_n 1
$M_n{}^a$ 2
n^A 5
Word[3]
ΔT_n^A 19LSB 19
$\Delta T_n'^A$ 7
ΔH_n^A 5
l_n 1
Word[4]
ω_n^A 8LSB 8
τ_λ^A 21
ΔT_n^A 3 MSB
Word[5]
ω_n^A 8MSB 8


Bit31 MSB
← 32Bits →
Bit0 LSB
Word[0]
$\Delta A^{\frac{1}{2}}$ 13
e 11
ω 8MSB 8
Word[1]
ω 8LSB 8
δ_i 11
Ω_0 13MSB 13
Word[2]
Ω_0 3LSB 3
$\dot{\Omega}$ 11
M_0 16
a_{f0} 2 MSB
Word[3]
a_{f0} 14 MSB 14
a_{f1} 13
E5b$_{HS}$ 2
E1B$_{HS}$ 2
Reserved 1


Bit31 MSB
← 32Bits →
Bit0 LSB
Word[0]
$\sqrt{A}$ 13
e 11
ω 8MSB 8
Word[1]
ω 8LSB 8
δ_i 11
Ω_0 13MSB 13
Word[2]
Ω_0 3LSB 3
$\dot{\Omega}$ 11
M_0 16
af0 2MSB 2
Word[3]
af0 14LSB 14
af1 13
E5a$_{HS}$ 2
Reserved 3





Return&Example

[Return]

1. PAIR_ACK for send result.

2. $PAIR551,<Status>*CS<CR><LF>

Status: 1, success; 0, fail.

[Example]

Send:

$PAIR551,0,0,01,080A,414956,24160B,FD6A00,A10CEA,775832,1D4992,0DEA80,FAFFA8*30\r\n

Response:

$PAIR001,551,0*3A\r\n ==> Success

$PAIR551,1*26\r\n

2.3.113 Packet Type:552 PAIR_ALM_GET_DATA

Get a single almanac subframe message.

DataField: $PAIR552,<Constellation>,<Signal_ID>,<SV_ID>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Constellation	--	--	The GNSS system ID: '0' = GPS '1' = GLONASS '2' = Galileo '3' = BeiDou '4' = QZSS
Signal_ID	--	--	Signal type L1: 0 Midi: 1 (including GPS L5, Galileo E5a, BeiDou B2a)
SV_ID			in DEC format GPS: 1-32 GLONASS: 1-24 Galileo: 1-36 BeiDou: 1-63 (GEO: 1-5, 59-63; MEO: 6-58) QZSS: 1-7

Return&Example

[Return]

1. PAIR_ACK for send result.

2. $PAIR552,<Constellation>,<Signal_ID>,<SV_ID>,<WeekNo>,<ALM_data>*CS<CR><LF>

ALM_data: refer to the format of PAIR551

[Example]

Send:

$PAIR552,0,0,1*25\r\n

Response:

$PAIR001,552,0*39\r\n ==> Success

$PAIR552,0,0,01,080A,414956,24160B,FD6A00,A10CEA,775832,1D4992,0DEA80,FAFFA8*33\r\n

2.3.114 Packet Type:553 PAIR_ALM_CLEAR

Clear the almanac data in the critical memory area

DataField: $PAIR553*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR553,*39\r\n
Response:
$PAIR001,553,0*38\r\n ==> Success

2.3.115 Packet Type:590 PAIR_TIME_SET_REF_UTC

Send current UTC time to GNSS chip for faster TTFF.
Please do not use local time which has a time-zone offset
For a faster TTFF, the accuracy of reference UTC is better if it is less than 3 seconds.

DataField: $PAIR590,<YYYY>,<MM>,<DD>,<hh>,<mm>,<ss>*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR591,<YYYY>,<MM>,<DD>,<hh>,<mm>,<ss>*CS<CR><LF>
YYYY year >= 2000 UTC time: year in 4 digits
MM month 1 - 12 UTC time: month
DD day 1 - 31 UTC time: day
hh hour 0 - 23 UTC time: hour
mm minute 0 - 59 UTC time: minute

ss second 0 - 59 UTC time: second
[Example]
Send:
$PAIR590,2019,2,10,9,0,58*0B\r\n
Response:
$PAIR001,590,0*37\r\n ==> Success
$PAIR590,2019,02,10,09,00,58*3B

2.3.116 Packet Type:591 PAIR_TIME_GET_REF_UTC

Query current UTC time set in GNSS chip

DataField: $PAIR591*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR591,<YYYY>,<MM>,<DD>,<hh>,<mm>,<ss>*CS<CR><LF>
YYYY year >= 2000 UTC time: year in 4 digits
MM month 1 - 12 UTC time: month
DD day 1 - 31 UTC time: day
hh hour 0 - 23 UTC time: hour
mm minute 0 - 59 UTC time: minute
ss second 0 - 59 UTC time: second
[Example]
Send:
$PAIR591*37\r\n
Response:
$PAIR001,591,0*36\r\n ==> Success
$PAIR591,2019,2,10,9,0,58*0A\r\n

2.3.117 Packet Type:592 PAIR_TIME_SET_UTC_CORRECTION_DATA

Set current UTC correction data.

DataField:	$PAIR592,<A1>,<A0>,<Tot>,<WNt>,<dtLS>,<WNLSF>,<DN>,<dtLSF>*CS<CR><LF>		
Name	**Unit**	**Default**	**Description**
A1	--	--	Constant terms of polynomial (2^-30 seconds)
A0			First order of polynomial (2^-50 seconds/second)
Tot			Tot reference time of week (2^12 seconds)
WNt			UTC reference week number
dtLS			Current or past leap second count (second)
WNLSF			Leap second reference week number
DN			Day number
dtLSF			Current or future leap second count (second)

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR592,7,2,0,64,18,137,7,18*01\r\n
Response:
$PAIR001,592,0*35\r\n ==> Success

2.3.118 Packet Type:593 PAIR_TIME_GET_UTC_CORRECTION_DATA

Query current UTC correction data.

DataField:	$PAIR593, *CS<CR><LF>		
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2.
$PAIR593,<Status>,<A1>,<A0>,<Tot>,<WNt>,<dtLS>,<WNLSF>,<DN>,<dtLSF>*CS<CR><LF>
Status: '1' means UTC correction data are available.
'0' means UTC correction data are not available.
when Status = '1', the following will be shown:
A1: Constant terms of polynomial (2^-30 seconds)

A0: First order of polynomial (2^-50 seconds/second)
Tot: Tot reference time of week (2^12 seconds)
WNt: UTC reference week number
dtLS: Current or past leap second count (second)
WNLSF: Leap second reference week number
DN: Day number
dtLSF: Current or future leap second count (second)

[Example]
Send:
$PAIR593*35\r\n
Response:
$PAIR001,593,0*34\r\n ==> Success
$PAIR593,1,7,2,0,64,18,137,7,18*1D

2.3.119 Packet Type:595 PAIR_TIME_CONVERT_TOW_FROM_32K_FREE_COUNT

Convert the free count from 32K clock source to the time of week in milliseconds.

DataField: $PAIR595,<Free_Count>*CS<CR><LF>

Name	Unit	Default	Description
Free_Count	--	--	Free count from 32K clock source in milliseconds (0 ~ 131071999)

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR595,<Validity>,<TOW>*CS<CR><LF>
Validity: TOW validity.
0: Invalid TOW.
1: Valid TOW.
TOW: Free_Count's corresponding time of week in milliseconds.

[Example]
Send:
$PAIR595,69053*26\r\n
Response:
$PAIR001,595,0*32\r\n ==> Success
$PAIR595,1,96710573*0A\r\n ==> Valid TOW (96710573) converted from Free_Count (69053)

2.3.120 Packet Type:596 PAIR_TIME_GET_CURRENT_TOW

Get current time of week in milliseconds.

DataField: $PAIR596*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR596,<Validity>,<TOW>*CS<CR><LF>
Validity: Validity of TOW.
0: Invalid TOW.
1: Valid TOW.
TOW: Current time of week in milliseconds.

[Example]
Send:
$PAIR596*30\r\n
Response:
$PAIR001,596,0*31\r\n ==> Success
$PAIR596,1,96939680*03\r\n ==> Valid current TOW

2.3.121 Packet Type:597 PAIR_TIME_GET_GNSS_TOW

Get the last GNSS epoch's time of week in milliseconds.

DataField: $PAIR597*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR597,<Validity>,<TOW>*CS<CR><LF>
Validity: Validity of TOW.
0: Invalid TOW.
1: Valid TOW.

TOW: The last GNSS epoch's time of week in milliseconds.
[Example]
Send:
$PAIR597*31\r\n
Response:
$PAIR001,597,0*30\r\n ==> Success
$PAIR597,1,96710000*09\r\n ==> Valid TOW of the last GNSS epoch

2.3.122 Packet Type:600 PAIR_LOC_SET_REF

Send reference location to GNSS chip for faster TTFF.

DataField:
$PAIR600,<Latitude>,<Longitude>,<Height>,<AccMaj>,<AccMin>,<Bear>,<AccVert>*CS<CR><LF>

Name	Unit	Default	Description
Latitude	--	--	reference latitude in degrees
Longitude			reference longitude in degrees
Height			reference height in meters
AccMaj			semi-major RMS accuracy [m]
AccMin			semi-minor RMS accuracy [m]
Bear			Bearing in degrees
AccVert			Vertical RMS accuracy [m]

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR600,24.772816,121.022636,175.0,50.0,50.0,0.0,100.0*06\r\n
Response:
$PAIR001,600,0*3D\r\n ==> Success

2.3.123 Packet Type:604 PAIR_LOC_SET_FIX_POSITION

Send stationary fix position for GNSS chip. In some case, the reciever's position must be known precisely.

This is for situations such as RTK Base Receiver, Timing position-hold mode

DataField: $PAIR604,<Enable>,<Mode>,<Para1>,<Para2>,<Para3>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	0: Disable 1: Enable fix position for GNSS chip.
Mode			0: Position in ECEF coordinate (XYZ) 1: Position in WGS84, (Lat,Lon,Height)
Para1			ECEF X (m) or Latitude (degrees)
Para2			ECEF Y (m) or Longitude (degrees)
Para3			ECEF Z (m) or Height (meter, over ellipsoid height)

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR604,1,1,24.7728165,121.0226365,175.01*32\r\n
Response:
$PAIR001,606,0*39\r\n ==> Success

2.3.124 Packet Type:605 PAIR_LOC_SET_FIX_POSITION

Get the stationary fix position setting from GNSS chip.

DataField: $PAIR605*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR605,<Enable>,<Mode>,<Para1>,<Para2>,<Para3>*CS<CR><LF>
Enable:
0: Disable
1: Enable fix position for GNSS chip.
Mode:
0: Position in ECEF coordinate (XYZ)

1: Position in WGS84, (Lat,Lon,Height)
Para1:
ECEF X (m) or Latitude (degrees)
Para2:
ECEF Y (m) or Longitude (degrees)
Para3:
ECEF Z (m) or Height (meter, over ellipsoid height)

[Example]

Send:

$PAIR605*39\r\n

Response:

$PAIR001,605,0*38\r\n ==> Success

$PAIR605,1,1,24.7728165,121.0226365,175.01*33\r\n

2.3.125 Packet Type:606 PAIR_LOC_ENABLE_PR_RESIDUALS_OUTPUT

Set the position and get the corresponding PR residuals

DataField: $PAIR606,<Enable>,<Latitude>,<Logitude>,<Height>*CS<CR><LF>

Name	Unit	Default	Description
Enable	--	--	0: Disable 1: Enable
Latitude			Latitude (unit: degree) - range: -90 ~ 90 degs
Logitude			Logitude (unit: degree) - range: -180 ~ 180 degs
Height			Ellipsoidal Height (unit: meter) - range: -30000000 ~ 30000000 m

Return&Example

[Return]

1. PAIR_ACK for send result.

[Example]

Send:

$PAIR606,1,0,0,0*3B\r\n

Response:

$PAIR001,606,0*3B\r\n ==> Success

Will also output the corresponding Pseudorange residuals. The detail is referred to the Message ID = 4006 of the document "Airoha_IoT_SDK_Location_Raw_Measurement_User_Guide".

NOTE

1. if this function is enabled and Lat=Lon=Hgt=0.0, it will output the original KF PR residuals;
2. if this function is enabled and one of the arguments (Lat\Lon\Hgt) is not equal to zero, it will output the PR residuals based on the inputted position;.

2.3.126 Packet Type:610 PAIR_HOTSTILL_ENABLE

Enable or disable the hotstill function.

DataField: $PAIR610,<Enable>*CS<CR><LF>

Name	Unit	Default	Description
Enable	--	--	Enable or disable '0': Disable '1': Enable

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR610,1*20\r\n
Response:
$PAIR001,610,0*3C\r\n

2.3.127 Packet Type:611 PAIR_HOTSTILL_NEW_EPH_NOTIFY

Send notification message (PAIR611,0) to host if new broadcast ephemeris is available.
Host should send PAIR611,1 to go to next stage.

DataField: $PAIR611,<Para1>*CS<CR><LF>

Name	Unit	Default	Description
Para1	--	--	notification message [Range: 0-1] '0': device to host '1': host to device

Return&Example

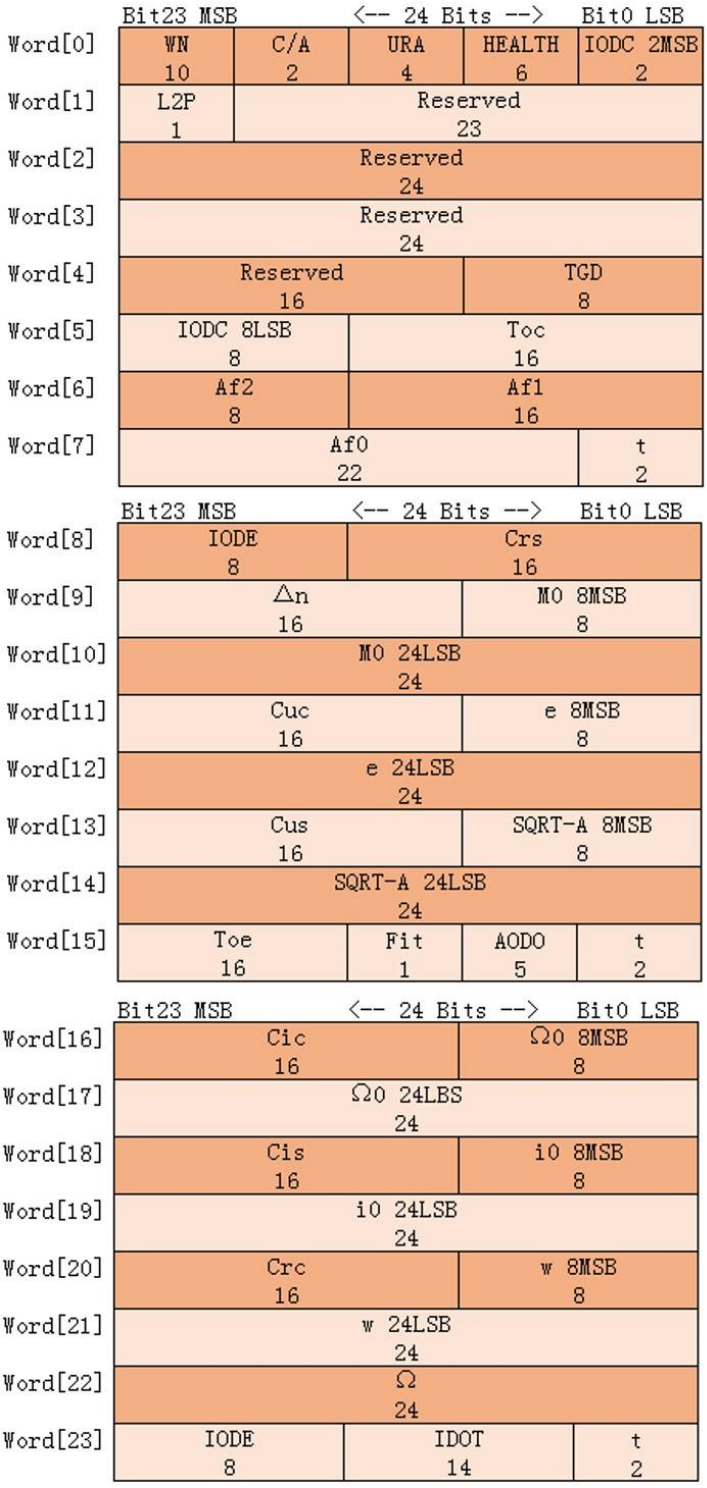
[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR611,1*21
Response:
$PAIR001,611,0*3D\r\n ==> Success

2.3.128 Packet Type:612 PAIR_HOTSTILL_INDICATION_NEW_EPH_DATA

Output GPS ephemeris data message to host.

DataField: $PAIR612,<SV_ID>,<W[0]>,...,<W[23]>*CS<CR><LF>

Name	Unit	Default	Description
SV_ID	--	--	in HEX format [Range: 1-20]
GPS(L1)			W[0],...,W[23] 24 words of the ephemeris subframes data from words 3 to 10 of subframes 1, 2 and 3 of the GPS Navigation Message. Each of the raw 30 bit data words have been logically shifted 6 bits to the right to remove the 6 parity bits leaving the 24 data bits.



Return&Example

[Example]
Response:

$PAIR612,0A,145000,3897C9,7E0AF9,E13E4A,5BAC05,2332FA,00FFAE,BFC87E,23F952,3493FB,61584B,F9E803,1AC578,0866A1,0D5186,32FA7E,001237,3E4CA4,002B27,624737,25F293,853024,FFA390,23F151*63

NOTE

The GNSS system automatically sends this command. Please do not actively send it to the GNSS system..

2.3.129 Packet Type:613 PAIR_HOTSTILL_NEW_EPH_ACK

Host received ephemeris by PAIR612, then send PAIR613 as ack....

DataField: $PAIR613,<SV_ID>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
SV_ID	--	--	in DEC format [Range: 1-32]

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR613,1*23
Response:
$PAIR001,613,0*3F

2.3.130 Packet Type:614 PAIR_HOTSTILL_REQ

Send hotstill request message to host.
Host received this message should send the hotstill data to aiding....

DataField: $PAIR614,<WeekNo>,<TOW>,<Num_SV>,<SV_ID_1>,<SV_ID_2>,...*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
WeekNo	--	--	GNSS Week number in DEC format
TOW			GNSS Time of week in DEC format
Num_SV			Number of satellites
SV_ID			in DEC format [Range: 1-32]

Return&Example

[Example]
Response:

$PAIR614,2118,186199,12,2,3,4,5,6,9,12,14,17,19,23,28*1E

2.3.131 Packet Type:615 PAIR_HOTSTILL_HOST_INFO

A list of satellites for which the host has their hotstill data...

DataField: $PAIR615,<Num_SV>,<SV_ID_1>,<SV_ID_2>,...*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Num_SV	--	--	words [LSB first] of one HotStill data (total 64 bytes)
SV_ID			in DEC format [Range: 1-32]

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR615,3,1,31,32*39
Response:
$PAIR001,615,0*39

2.3.132 Packet Type:616 PAIR_HOTSTILL_DATA

Send the Hotstill data for a single satellite..

DataField: $PAIR616,<W[0]>,...,<W[15]>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
W[0] ~ W[15]	--	--	words [LSB first] of one HotStill data (total 64 bytes)

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR616,CBDFA336,00FA1559,D4CA0D07,0214E530,00BD0ACE,B627F6FF,16AD0818,6A965D3D,64A16F4C,800D0117,00000000,0000004B,00000000,00000000,00000000,00000000*40
Response:

$PAIR001,616,0*3A

2.3.133 Packet Type:617 PAIR_HOTSTILL_INDICATION_END_DATA_ACK

End of hotstill data transmission message from Host(PAIR614,-1).
Send this command as ACK.

DataField: $PAIR617,<Para1>*CS<CR><LF>

Name	Unit	Default	Description
Para1	--	--	notification message [Range: 0] '0': device to host

Return&Example

[Example]
Response:
$PAIR617,0*26

NOTE

The GNSS system automatically sends this command. Please do not actively send it to the GNSS system.

2.3.134 Packet Type:618 PAIR_HOTSTILL_INDICATION_EPH_INFO

Output message to show available GPS ephemeris for individual satellite (used broadcast ephemeris or aiding data)

DataField: $PAIR618,<Num_SV>,<SV_ID_1>,<SV_ID_2>,...*CS<CR><LF>

Name	Unit	Default	Description
Num_SV	--	--	Number of satellites
SV_ID			in DEC format [Range: 1-32]

Return&Example

[Example]
Response:
$PAIR618,15,1,2,3,4,5,6,8,9,11,12,17,19,22,28,30*33

NOTE

The GNSS system automatically sends this command. Please do not actively send it to the GNSS system..

2.3.135 Packet Type:650 PAIR_LOW_POWER_ENTRY_RTC_MODE

Shutdown all systems, including GNSS and other CM4 modules
CM4 will go into RTC-Mode after sending this command and cannot receive any commands. CM4 can be awoken by the timer or the RTC_EINT pin. All system resource will re-initialize after wake up

DataField: $PAIR650,<Second>*CS<CR><LF>

Name	Unit	Default	Description
Second	--	--	the timer to leave RTC-Mode [Valid range: 0 and 10 ~ 62208000 (2 years)] '0' enter RTC-Mode without any timer

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR650,1*24\r\n
Response:
$PAIR001,650,4*3C\r\n ==> Parameter error
Send:
$PAIR650,10*14\r\n
Response:
Enter RTC-Mode without any response and wake up after 10 seconds

NOTE

S/SD EVK must require timer to enter RTC-Mode..

2.3.136 Packet Type:680 PAIR_GLP_ENABLE

This command is to activate low-power GLP mode.
GLP mode supports 1-Hz PVT, GPS L1 only, and Fitness mode.

DataField: $PAIR680,<Enable>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	0: Disable GLP 1: Enable GLP

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR680,1*29\r\n
Response:
$PAIR001,680,0*35\r\n ==> Success

2.3.137 Packet Type:681 PAIR_GLP_GET_STATUS

This command is to get low-power GLP mode setting.

DataField: $PAIR681*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
	--	--	

Return&Example

[Return]
1. PAIR_ACK for send result.

2. $PAIR681,<Enable>*CS<CR><LF>
Enable:
0: Disable GLP
1: Enable GLP.
[Example]
Send:
$PAIR681*35\r\n
Response:
$PAIR001,681,0*34\r\n ==> Success
$PAIR681,1*28\r\n

2.3.138 Packet Type:690 PAIR_PERIODIC_SET_MODE

This command is used to set Periodic Power Saving Mode Settings.
There are two stages in periodic power saving mode (Run stage and Sleep stage), and it will change periodically according to the setting.
Run stage: the GNSS module measures and calculates the position.
Sleep stage: the GNSS module may enter power saving modes.
<Note> Sleep will be interrupted by any DSP corresponding PAIR command.
Any restart will force it to go back to normal mode.
For more detailed information, please refer to the Power Saving Mode chapter of Periodic Mode section in the Airoha_IoT_SDK_for_GNSS_Developers_Guide under the doc folder in IoT_SDK_for_Location package.

DataField:
$PAIR690,<Mode>,<FirstRun>,<FirstSleep>,<SecondRun>,<SecondSleep>*CS<CR><LF>

Name	Unit	Default	Description
Mode	--	--	0: Disable periodic mode 1: Smart periodic mode. In this mode, GNSS system dynamically increases run time in order to collect more navigation data 2: Strict periodic mode. In this mode, GNSS system periodically forces entry into low-power mode If <Mode> is 1 or 2, it needs the following parameter for low-power periodic mode
FirstRun	--	--	Interval in seconds to exit the minimum power sleep mode and get a new position fix. [Range: 3~518400 s]
FirstSleep	--	--	Duration in seconds to get a fix (or attempt to get a fix) before switching from running mode back to a minimum power sleep mode. [Range: 3~518400 s]
SecondRun	--	--	GNSS system will use "second run time" instead of "run time" setting when there is no signal. [Range: 0 or 3~518400 s] The

			second run time duration can be "0" only when the second sleep time is "0"
SecondSleep	--	--	GNSS system will use "second sleep time" instead of "sleep time" setting when there is no signal. [Range: 0 or 3~518400 s] The second sleep time duration can be "0" only when the second run time is "0"

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR690,1,21,39,48,72*28\r\n
Response:
$PAIR001,690,0*34\r\n ==> Success
Send:
$PAIR690,0*29\r\n ==> Normal mode
Response:
$PAIR001,690,0*34\r\n ==> Success

2.3.139 Packet Type:691 PAIR_PERIODIC_GET_MODE

This command is used to get Periodic Power Saving Mode Settings.
For more detailed information, please refer to the Power Saving Mode chapter of Periodic Mode section in the
Airoha_IoT_SDK_for_GNSS_Developers_Guide under the doc folder in IoT_SDK_for_Location package

DataField:	$PAIR691*CS<CR><LF>		
Name	**Unit**	**Default**	**Description**
	--	--	

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR691,<Mode>,<FirstRun>,<FirstSleep>,<SecondRun>,<SecondSleep>*CS<CR><LF>
Mode:
0: Disable periodic mode.
1: Smart periodic mode. In this mode, GNSS system dynamically increases run time in order to collect more navigation data.

2: Strict periodic mode. In this mode, GNSS system periodically forces entry into low-power mode.

FirstRun: Interval in seconds to exit the minimum power sleep mode and get a new position fix. [Range: 3~518400 s]

FirstSleep: Duration in seconds to get a fix (or attempt to get a fix) before switching from running mode back to a minimum power sleep mode. [Range: 3~518400 s]

SecondRun: GNSS system will use "second run time" instead of "run time" setting when there is no signal. [Range: 0 or 3~518400 s]

SecondSleep: GNSS system will use "second sleep time" instead of "sleep time" setting when there is no signal. [Range: 0 or 3~518400 s].

[Example]

Send:

$PAIR691*34\r\n

Response:

$PAIR001,691,1*34\r\n

$PAIR691,1,21,39,48,72*29\r\n

Send:

$PAIR691*34\r\n

Response:

$PAIR001,691,1*34\r\n

$PAIR691,0,21,39,48,72*28\r\n ==> Normal mode

2.3.140 Packet Type:750 PAIR_PPS_SET_CONFIG

Set the configuration of the local time in milliseconds and phase where the PPS should be placed

DataField: $PAIR750,<PPS_by_user>,<Local_ms>,<Phase>*CS<CR><LF>

Name	Unit	Default	Description
PPS_by_user	--	--	"1", PPS output by user "0", PPS automatic output
Local_ms	--	--	Local receiver time tick. Range is from 0 to 4294967295 (232-1). If PSS is enabled, this parameter aligns to TOW
Phase	--	--	Time tick phase range is from 0 to 262143. If PSS is enabled, this parameter aligns to TOW

Return&Example

[Return]

1. PAIR_ACK for send result

[Example]

Send:

$PAIR750,1,1345,555*13\r\n
Response:
$PAIR001,750,0*39\r\n ==> Success

2.3.141 Packet Type:752 PAIR_PPS_SET_CONFIG_CMD

Configure the PPS settings

DataField: $PAIR752,<PPSType>,<PPSPulseWidth>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
PPSType	--	--	Availability "0", Disable "1", After the first fix "2", 3D fix only "3", 2D/3D fix only "4", Always
PPSPulseWidth	--	--	PPS Pulse Width (unit in ms). [Range: 1 ~ 999].

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR752,2,100*39\r\n
Response:
$PAIR001,752,0*3B\r\n ==> Success

2.3.142 Packet Type:753 PAIR_PPS_SET_TIMING_PRODUCT

The timing product mode will enhance the PPS output timing accuracy.
mode1: For getting higher timing accuracy, SBAS/QZSS effects are disabled.

DataField: $PAIR753,<Timing Product>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Timing Product	--	--	'0': Disable. '1': Enable timing product (remove SBAS/QZSS effects)

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR753,1*26\r\n
Response:
$PAIR001,753,0*3A\r\n ==> Success

NOTE

Please measure the accuracy after the device collects all of the satellite almanac data

2.3.143 Packet Type:755 PAIR_PPS_SET_TIMETAG

Set enable/disable output time tag and time base.

DataField: $PAIR755,<Enable>,<Time_base>*CS<CR><LF>

Name	Unit	Default	Description
Enable	--	--	"0", Disable. "1", Enable.
Time_base	--	--	(Now only support GPS time base) "0", UTC. "1", GPS. "2", GLO. "3", GAL. "4", BDS. "5", NavIC.

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR755,1,1*3D\r\n
Response:

$PAIR001,755,0*3C\r\n ==> Success

2.3.144 Packet Type:756 PAIR_PPS_GET_TIMETAG_CONFIG

Get time tag configuration including output status and time base.

DataField: $PAIR756*CS<CR><LF>

Name	Unit	Default	Description
Enable	--	--	0: disable 1: raw meas 2: raw meas + sv info + pvt (including time offset data between GPS and GLO/GAL/BDS)

Return&Example

[Return]
$PAIR756,<Enable>,<Time_base>*CS<CR><LF>
Enable:
"0", Disable.
"1", Enable.
Time_base: (Now only support GPS time base)
"0", UTC.
"1", GPS.
"2", GLO.
"3", GAL.
"4", BDS.
"5", NavIC..
[Example]
Send:
$PAIR756*3E\r\n.
Response:
$PAIR001,756,0,1*22\r\n ==> Success

2.3.145 Packet Type:830 PAIR_RAW_ENABLE

Set enable/disable output binary raw measurement

DataField: $PAIR830,<Enable>*CS<CR><LF>

Name	Unit	Default	Description
Enable	--	--	0: disable 1: raw meas 2: raw meas + sv info + pvt (including time offset data between GPS and GLO/GAL/BDS)

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR830,1*2C\r\n.
Response:
$PAIR001,830,0*30\r\n ==> Success

2.3.146 Packet Type:831 PAIR_RAW_GET_STATUS

Get enable/disable output binary raw measurement

DataField: $PAIR831*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
	--	--	

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR831,<Enable>*CS<CR><LF>
Enable:
0: disable
1: raw meas
2: raw meas + sv info + pvt
[Example]
Send:
$PAIR831*30\r\n
Response:
$PAIR001,831,0*31\r\n ==> Success
$PAIR831,1*2D\r\n

2.3.147 Packet Type:860 PAIR_IO_OPEN_PORT

Open a GNSS data port

DataField:
$PAIR860,<Port_Type>,<Port_Index>,<Data_Type>,<Baudrate>,<Flow_control>*CS<CR><LF>

Name	Unit	Default	Description
Port_Type	--	--	HW Port Type: 0: UART [ER1 support] 1: I2C [ER2 support] 2: SPI [ER2 support] 3: USB [ER1 support] 4: SD-Card [ER3 support]
Port_Index	--	--	HW Port Index: UART - 0: UART0, 1: UART1, 2: UART2 USB - 0: USB Virtual Port 0, 1: USB Virtual Port 1 Others - 0: Only one port
Data_Type	--	--	A bitmap to config data type: GNSS_IO_FLAG_OUT_NMEA (0x01) GNSS_IO_FLAG_OUT_LOG (0x02) GNSS_IO_FLAG_OUT_CMD_RSP (0x04) GNSS_IO_FLAG_OUT_DATA_RSP (0x08) GNSS_IO_FLAG_OUT_RTCM (0x10) GNSS_IO_FLAG_IN_CMD (0x20) GNSS_IO_FLAG_IN_DATA (0x40) GNSS_IO_FLAG_IN_RTCM (0x80)
Baudrate	--	--	the baud rate must be configured. This parameter is only valid for UART. Please use 0 for other port type: Support 110, 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600, 3000000
Flow_control	--	--	0, disable flow control. 1, enable SW flow control. 2, enable HW flow control. This parameter is only valid for UART. Please use 0 for other port type

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR860,0,2,37,115200,0*29\r\n ==> Open UART2 to NMEA output without flow control. Baudrate is 115200.
Response:

$PAIR001,860,0*35\r\n ==> Success

2.3.148 Packet Type:861 PAIR_IO_CLOSE_PORT

Close a GNSS data port.

DataField: $PAIR861,<Port_Type>,<Port_Index>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Port_Type	--	--	HW Port Type: 0: UART [ER1 support] 1: I2C [ER2 support] 2: SPI [ER2 support] 3: USB [ER1 support] 4: SD-Card [ER3 support]
Port_Index	--	--	HW Port Index: UART - 0: UART0, 1: UART1, 2: UART2 USB - 0: USB Virtual Port 0, 1: USB Virtual Port 1 Others - 0: Only one port

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR861,0,2*37\r\n ==> Close UART2
Response:
$PAIR001,861,0*34\r\n ==> Success

NOTE

GNSS_IO_FLAG_IN_RTCM cannot be set with a different type in the same port

2.3.149 Packet Type:862 PAIR_IO_SET_DATA_TYPE

Set GNSS port data type configuration

DataField: $PAIR862,<Port_Type>,<Port_Index>,<Data_Type>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Port_Type	--	--	HW Port Type: 0: UART [ER1 support] 1: I2C [ER2 support] 2: SPI [ER2 support] 3: USB [ER1 support] 4: SD-Card [ER3 support]
Port_Index	--	--	HW Port Index: UART - 0: UART0, 1: UART1, 2: UART2 USB - 0: USB Virtual Port 0, 1: USB Virtual Port 1 Others - 0: Only one port
Data_Type	--	--	A bitmap to config data type: GNSS_IO_FLAG_OUT_NMEA (0x01) GNSS_IO_FLAG_OUT_LOG (0x02) GNSS_IO_FLAG_OUT_CMD_RSP (0x04) GNSS_IO_FLAG_OUT_DATA_RSP (0x08) GNSS_IO_FLAG_OUT_RTCM (0x10) GNSS_IO_FLAG_IN_CMD (0x20) GNSS_IO_FLAG_IN_DATA (0x40) GNSS_IO_FLAG_IN_RTCM (0x80).

Return&Example

[Return]

1. PAIR_ACK for send result.

[Example]

Send:

$PAIR862,3,1,37*1C\r\n ==> Config USB virtual port 1 to NMEA & PAIR port. (Without debug log.)

Response:

$PAIR001,862,0*37\r\n ==> Success

NOTE

GNSS_IO_FLAG_IN_RTCM cannot be set with a different type in the same port

2.3.150 Packet Type:863 PAIR_IO_GET_DATA_TYPE

Get GNSS port data type configuration

DataField: $PAIR863,<Port_Type>,<Port_Index>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Port_Type	--	--	HW Port Type: 0: UART [ER1 support] 1: I2C [ER2 support] 2: SPI [ER2 support] 3: USB [ER1 support] 4: SD-Card [ER3 support]
Port_Index	--	--	HW Port Index: UART - 0: UART0, 1: UART1, 2: UART2 USB - 0: USB Virtual Port 0, 1: USB Virtual Port 1 Others - 0: Only one port

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR863,<Data_Type>*CS<CR><LF>
Data_Type: A bitmap to config data type
GNSS_IO_FLAG_OUT_NMEA (0x01)
GNSS_IO_FLAG_OUT_LOG (0x02)
GNSS_IO_FLAG_OUT_CMD_RSP (0x04)
GNSS_IO_FLAG_OUT_DATA_RSP (0x08)
GNSS_IO_FLAG_OUT_RTCM (0x10)
GNSS_IO_FLAG_IN_CMD (0x20)
GNSS_IO_FLAG_IN_DATA (0x40)
GNSS_IO_FLAG_IN_RTCM (0x80)
[Example]
Send:
$PAIR863,3,1*35\r\n
Response:
$PAIR001,863,0*36\r\n ==> Success
$PAIR863,37*1F\r\n ==> Get USB Port1 data config is 37.
37--> 100101
--> GNSS_IO_FLAG_OUT_NMEA | GNSS_IO_FLAG_OUT_CMD_RSP | GNSS_IO_FLAG_IN_CMD

2.3.151 Packet Type:864 PAIR_IO_SET_BAUDRATE

Set port baud rate configuration

DataField: $PAIR864,<Port_Type>,<Port_Index>,<Baudrate>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Port_Type	--	--	HW Port Type: 0: UART [ER1 support]
Port_Index	--	--	HW Port Index: 0: UART0 1: UART1 2: UART2
Baudrate	--	--	the baud rate need config: Support 115200, 230400, 460800, 921600, 3000000

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR864,0,0,115200*1B\r\n
Response:
$PAIR001,864,0*31\r\n ==> Success

NOTE

Must reboot the device after changing the port baud rate. The change will valid after reboot

2.3.152 Packet Type:865 PAIR_IO_GET_BAUDRATE

Get port baud rate configuration

DataField: $PAIR865,<Port_Type>,<Port_Index>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Port_Type	--	--	HW Port Type: 0: UART [ER1 support]
Port_Index	--	--	HW Port Index: 0: UART0 1: UART1 2: UART2

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR865,<Baudrate>*CS<CR><LF>
Baudrate: the baud rate need config
Support 115200, 230400, 460800, 921600, 3000000
[Example]
Send:
$PAIR865,0,0*31\r\n
Response:
$PAIR001,865,0*30\r\n ==> Success
$PAIR865,115200*1A\r\n ==> Get UART0 baud rate is 115200

NOTE

Must reboot the device after changing the port baud rate

2.3.153 Packet Type:866 PAIR_IO_SET_FLOW_CONTROL

Set port flow control configuration.

DataField: $PAIR866,<Port_Type>,<Port_Index>,<Flow_control>*CS<CR><LF>

Name	Unit	Default	Description
Port_Type	--	--	HW Port Type. 0: UART
Port_Index	--	--	HW Port Index UART - 0: UART0, 1: UART1, 2: UART2
Flow_control			0, disable flow control. 1, enable SW flow control. 2, enable HW flow control.

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR866,0,2,1*2D\r\n ==> Set UART2 SW Flow Control ON
Response:

$PAIR001,866,0*33\r\n ==> Success

NOTE

Must reboot the device after changing the flow control type. The change will valid after reboot.

2.3.154 Packet Type:867 PAIR_IO_GET_FLOW_CONTROL

Get port SW flow control configuration.

DataField: $PAIR867,<Port_Type>,<Port_Index>*CS<CR><LF>

Name	Unit	Default	Description
Port_Type	--	--	HW Port Type. 0: UART
Port_Index	--	--	HW Port Index UART - 0: UART0, 1: UART1, 2: UART2

Return&Example

[Return]

2. PAIR_ACK for send result

2. $PAIR867,<sw_flow_control>*CS<CR><LF>

Flow_control: 0, disable flow control. 1, enable SW flow control. 2, enable HW flow control.

[Example]

Send:

$PAIR867,0,2*31\r\n

Response:

$PAIR001,867,0*32\r\n

$PAIR867,0*2F\r\n ==> Get UART2 Flow Control OFF

2.3.155 Packet Type:870 PAIR_IO_TEST

Check if PAIR channel is ready to work.

DataField: $PAIR870*CS<CR><LF>

Name	Unit	Default	Description

	--	--	

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR870*35\r\n
Response:
$PAIR001,870,0*34\r\n

2.3.156 Packet Type:890 PAIR_GEOFENCE_SET_CONFIG

This command is used to set Geofencing configuration.

DataField:
$PAIR890,<FenceNum>,<ConfLvl>,<Lat1>,<Lon1>,<Rad1>,...,<RadN>*CS<CR><LF>

Name	Unit	Default	Description
FenceNum(N)	--	--	Number of geofences, the value is limited to 4. When the value is set to 0, the geofencing function is disabled.
ConfLvl			The confidence level for state evaluation. '0' No requirement '1' 1-Sigma (68%) '2' 2-Sigma (95%) '3' 3-Sigma (99.7%) '4' 4-Sigma (99.9999%) '5' 4-Sigma (99.999999%)
Lat			Latitude of the geofence circle center (deg)
Lon			Longitude of the geofence circle center (deg)
Rad			Radius of the geofence circle (m)

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Enable the geofencing function:
Send:
$PAIR890,1,1,25.0567,121.5743,30*20\r\n

Response:
$PAIR001,890,0*3A\r\n ==> Success
Disable the geofencing function:
Send:
$PAIR890,0*27\r\n
Response:
$PAIR001,890,0*3A\r\n ==> Success

2.3.157 Packet Type:891 PAIR_GEOFENCE_GET_CONFIG

This command is used to get Geofencing configuration..

DataField: $PAIR891*CS<CR><LF>			
Name	Unit	Default	Description
	--	--	

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR891,<FenceNum>,<ConfLvl>,<Lat1>,<Lon1>,<Rad1>,...,<RadN>*CS<CR><LF>
FenceNum(N): Number of geofences, the value is limited to 4.
ConfLvl: The confidence level for state evaluation.
'0' No requirement
'1' 1-Sigma (68%)
'2' 2-Sigma (95%)
'3' 3-Sigma (99.7%)
'4' 4-Sigma (99.9999%)
'5' 4-Sigma (99.999999%)
Lat: Latitude of the geofence circle center (deg)
Lon: Longitude of the geofence circle center (deg)
Rad: Radius of the geofence circle (m)
[Example]
Send:
$PAIR891*3A\r\n
Response:
$PAIR001,891,0*3B\r\n ==> Success
$PAIR891,1,1,25.0567,121.5743,30*21\r\n

2.3.158 Packet Type:892 PAIR_GEOFENCE_SET_GPIO_POLARITY

This command is used to set GPIO polarity for geofencing combined state.

DataField: $PAIR892,<GPIOPolarity>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
GPIOPolarity	--	--	GPIOPolarity: Pin polarity. '0' Low means outside '1' Low means inside Unknown state is always High.

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR892,1*24\r\n
Response:
$PAIR001,892,0*38\r\n ==> Success

2.3.159 Packet Type:900 PAIR_LOCUS_ENABLE

Enable or disable LOCUS save data

DataField: $PAIR900,<Enable>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	Enable: Enable or disable '0': Disable (Default) '1': Enable

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR900,1*2E\r\n ==> Enable LOCUS

Response:
$PAIR001,900,0*32\r\n ==> Enable Success

2.3.160 Packet Type:901 PAIR_LOCUS_GET_STATUS

Get LOCUS status

DataField: $PAIR901*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Enable	--	--	Enable: Enable or disable '0': Disable (Default) '1': Enable

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR901,<Enable>*CS<CR><LF>
Enable: Enable or disable
'0': Disable
'1': Enable
[Example]
Send:
$PAIR901*32\r\n
Response:
$PAIR001,901,0*33\r\n
$PAIR901,0*2E\r\n ==> LOCUS is disable

2.3.161 Packet Type:902 PAIR_LOCUS_SET_MODE

Set LOCUS saving mode

DataField: $PAIR902,<Mode>,<Check_3D_Fix>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Mode	--	--	Mode: Saving Mode: Normal, (1 << 0). Record per fix Out of time, (1 << 1). Record every N s. N is customer

			configuration (PAIR_LOCUS_SET_THRESHOLD) Out of speed, (1 << 2). Record after speed more than N m/s. N is customer configuration (PAIR_LOCUS_SET_THRESHOLD) Out of distance, (1 << 3). Record after distance more than N m. N is customer configuration (PAIR_LOCUS_SET_THRESHOLD) Before entry sleep, (1 << 4). Record before entry sleep User control, (1 << 5). Record after user send PAIR_LOCUS_LOG_NOW
Check_3D_Fix	--	--	Need check 3D fix or not: 0: not check 1: need check. If set this type as 1, system will not save the location without 3D fixed

Return&Example

[Return]
1. PAIR_ACK for send result
[Example]
Send:
$PAIR902,6,1*36\r\n ==> Set mode as out of time & out of speed mode. Need check 3D fix.
Response:
$PAIR001,902,0*30\r\n ==> Set success

NOTE

Must disable LOCUS saving before send this command

2.3.162 Packet Type:903 PAIR_LOCUS_GET_MODE

Get LOCUS saving mode

DataField: $PAIR903*CS<CR><LF>

Name	Unit	Default	Description
Mode	--	--	Mode: Saving Mode: Normal, (1 << 0). Record per fix Out of time, (1 << 1). Record every N s. N is customer configuration (PAIR_LOCUS_SET_THRESHOLD) Out of speed, (1 << 2). Record after speed more than N m/s. N

			is customer configuration (PAIR_LOCUS_SET_THRESHOLD) Out of distance, (1 << 3). Record after distance more than N m. N is customer configuration (PAIR_LOCUS_SET_THRESHOLD) Before entry sleep, (1 << 4). Record before entry sleep User control, (1 << 5). Record after user send PAIR_LOCUS_LOG_NOW
Check_3D_Fix	--	--	Need check 3D fix or not: 0: not check 1: need check. If set this type as 1, system will not save the location without 3D fixed

Return&Example

[Return]

1. PAIR_ACK for send result.

2. $PAIR903,<Mode>,<Check_3D_Fix>*CS<CR><LF>

Mode: Saving Mode

Normal, (1 << 0). Record per fix.

Out of time, (1 << 1). Record every N s. N is customer configuration (PAIR_LOCUS_SET_THRESHOLD).

Out of speed, (1 << 2). Record after speed more than N m/s. N is customer configuration (PAIR_LOCUS_SET_THRESHOLD).

Out of distance, (1 << 3). Record after distance more than N m. N is customer configuration (PAIR_LOCUS_SET_THRESHOLD).

Before entry sleep, (1 << 4). Record before going to sleep.

User control, (1 << 5). Record after user send PAIR_LOCUS_LOG_NOW.

Check_3D_Fix: Need check 3D fix or not.

0: not check.

1: need check. If set this type as 1, system will not save the location without 3D fixed.

[Example]

Send:

$PAIR903*30\r\n

Response:

$PAIR001,903,0*31\r\n

$PAIR903,6,1*37\r\n ==> LOCUS saving mode is out of time & out of speed mode. Need check 3D fix

NOTE

Must disable LOCUS saving before send this command

2.3.163 Packet Type:904 PAIR_LOCUS_SET_THRESHOLD

Set LOCUS mode threshold

DataField: $PAIR904,<Mode>,<Threshold>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Mode	--	--	Saving Mode: 0: Out of time mode 1: Out of speed mode 2: Out of distance mode
Threshold	--	--	The threshold of saving mode: If mode == 0, out of time mode, the time threshold is 1s ~ 12hours. Unit is second. Default is 15s If mode == 1, out of speed mode, the speed threshold is 1m/s ~ 100m/s. Unit is meter/secode. Default is 1m/s If mode == 2, out of distance mode, the distance threshold is 1m ~ 50000m. Unit is meter. Default is 1m

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR904,1,5*33\r\n ==> Set out of time mode threshold is 5s.
Response:
$PAIR001,904,0*36\r\n ==> Set success. LOCUS will save record every 5s.

NOTE

Must disable LOCUS saving before send this command
If the threshold out of rang, will response parameter error ("$PAIR001,804,4*33\r\n")

2.3.164 Packet Type:905 PAIR_LOCUS_GET_THRESHOLD

Get LOCUS mode threshold

DataField: $PAIR905,<Mode>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Mode	--	--	Saving Mode: 0: Out of time mode 1: Out of speed mode 2: Out of distance mode
Threshold	--	--	The threshold of saving mode: If mode == 0, out of time mode, the time threshold is 1s ~ 12hours. Unit is second. Default is 15s If mode == 1, out of speed mode, the speed threshold is 1m/s ~ 100m/s. Unit is meter/secode. Default is 1m/s If mode == 2, out of distance mode, the distance threshold is 1m ~ 50000m. Unit is meter. Default is 1m

Return&Example

[Return]

1. PAIR_ACK for send result

2. $PAIR905,<Threshold>*CS<CR><LF>

Threshold: The threshold of saving mode

If mode == 0, out of time mode, the time threshold is 1s ~ 12hours. Unit is second. Default is 15s

If mode == 1, out of speed mode, the speed threshold is 1m/s ~ 100m/s. Unit is meter/secode. Default is 1m/s

If mode == 2, out of distance mode, the distance threshold is 1m ~ 50000m. Unit is meter. Default is 1m

[Example]

Send:

$PAIR905,0*2A\r\n ==> Get time threshold

Response:

$PAIR001,905,0*37\r\n

$PAIR905,15*1E\r\n ==> Time threshold is 15s

NOTE

Must disable LOCUS saving before send this command

2.3.165 Packet Type:906 PAIR_LOCUS_CLEAR

Clear LOCUS Data

DataField: $PAIR906,<Type>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Type	--	--	Clear Type: 0: Clear record data and restore to default setting (configuration in gnss_config.bin) 1: Clear record data only 2: Clear user setting. Restore to default setting

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR906,0*29\r\n
Response:
$PAIR001,906,0*34\r\n

NOTE

Must disable LOCUS saving before send this command

2.3.166 Packet Type:907 PAIR_LOCUS_LOG_NOW

Save current location data

DataField: $PAIR907*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
	--	--	

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:

$PAIR907*34\r\n
Response:
$PAIR001,907,0*35\r\n

NOTE

Must keep user control (1 << 5) in saving mode if need use this command

2.3.167 Packet Type:908 PAIR_LOCUS_GET_DATA

Get all record data

DataField: $PAIR908,<Type>*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Type	--	--	Type: Response type: 0: Response as NMEA. 1: Response as PAIR command.

Return&Example

[Return]
1. PAIR_ACK for send result
2. $PAIR908,0*CS<CR><LF>
   LOCUS read begin
3. $PAIR908,1,<Record_Num>,<Record_Size>*CS<CR><LF>
   LOCUS read information
   Record_Num: the total record numbers
   Record_Size: the size of data per record
4. LOGGA + LORMC
   If type is 0, system will response LOGGA + GPGGA. The format is same as GPGGA + GPRMC.
5. $PAIR908,2,<UTC>,<Fix_Type>,<Lat>,<Lon>,<Heighing>,<Speed>,<Heading>,<HDOP>,<SatNo>*CS<CR><LF>
   If type is 1, system will response PAIR908,2,xxxx list for every record
   None saved data will show 0.
6. $PAIR908,3*CS<CR><LF>
   LOCUS read end

[Example]

Send:

$PAIR908,0*27\r\n

Response:

$PAIR001,908,0*3A\r\n
$PAIR908,0*27\r\n
$PAIR908,1,2,16*13\r\n
$LOGGA,080931.000,011772.4267,N,0016183.7702,E,1,0,0.0,0.53,M,,M,,*59\r\n
$LORMC,080931.000,A,011772.4267,N,0016183.7702,E,260320,,,A,V*C\r\n
$LOGGA,080932.000,011772.4267,N,0016183.7702,E,1,0,0.0,0.53,M,,M,,*5A\r\n
$LORMC,080932.000,A,011772.4267,N,0016183.7702,E,260320,,,A,V*F\r\n
$PAIR908,3*24\r\n
$PAIR001,908,0*3A\r\n

5	$PAIR908,0*27\r\n
6	$PAIR908,1,2,16*13\r\n
7	$LOGGA,080931.000,011772.4267,N,0016183.7702,E,1,0,0.0,0.53,M,,M,,*59\r\n
8	$LORMC,080931.000,A,011772.4267,N,0016183.7702,E,260320,,,A,V*C\r\n
9	$LOGGA,080932.000,011772.4267,N,0016183.7702,E,1,0,0.0,0.53,M,,M,,*5A\r\n
10	$LORMC,080932.000,A,011772.4267,N,0016183.7702,E,260320,,,A,V*F\r\n
11	$PAIR908,3*24\r\n

Send:

$PAIR908,1*26\r\n

Response:

$PAIR001,908,0*3A\r\n
$PAIR908,0*27\r\n
$PAIR828,2,5EA541BB,01,12341A1C,3E06BA8C,0210,0000,0000,0000,00*07r\n
$PAIR828,2,5EA541BC,01,12341A1B,3E06BA8A,0210,0000,0000,0000,00*05r\n
$PAIR908,1,2,16*13\r\n
$PAIR908,3*24\r\n

NOTE

Must disable LOCUS saving before send this command

2.3.168 Packet Type:909 PAIR_LOCUS_GET_RECORD_NUM

Get total record number

DataField: $PAIR909*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
Time	msec	--	Position fix interval in milliseconds (ms)

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR909,<Record_Num>*CS<CR><LF>
Record_Num: total record number
[Example]
Send:
$PAIR909*3A\r\n
Response:
$PAIR001,909,0*3B\r\n
$PAIR909,15*12\r\n ==> LOCUS has save 15 records

2.3.169 Packet Type:920 PAIR_BATCHING_ENABLE

Enable/Disable batching function.

DataField: $PAIR920,<Enable>*CS<CR><LF>

Name	Unit	Default	Description
Enable:	--	--	0: Disable (Default) 1: Enable

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR920,1*2C\r\n ==> enable batching feature
Response:
$PAIR001,920,0*30\r\n ==> Success

2.3.170 Packet Type:921 PAIR_BATCHING_GET_STATUS

Get batching status and recorded number.

DataField: $PAIR921*CS<CR><LF>			
Name	Unit	Default	Description
--	--	--	--

Return&Example

[Return]

2. PAIR_ACK for send result.

3. $PAIR921,<Enable>,<Record number>*CS<CR><LF>

Enable:

0: Disable

1: Enable

Record number: The number that already recorded.

[Example]

Send:

$PAIR921*30\r\n

Response:

$PAIR001,921,0*31\r\n ==> Success

$PAIR921,1,5*34\r\n ==> Batching function is enable, and there are 5 epoch recorded.

2.3.171 Packet Type:922 PAIR_BATCHING_SET_CONFIGURATION

Set baching configuration.

DataField: $PAIR922,<IntervalThres>,<DistThres>,<SpdThres>,<DataFormat>*CS<CR><LF>			
Name	Unit	Default	Description
IntervalThres(s)	--	--	If the time interval of recording location, batching will record the location. The interval threshold is 1s ~ 12hours. Unit is second. Default is 1s.
DistThres(m)			When the distance between current and previous location exceed this value, batching will record the location. The distance threshold is 1m ~ 50000m. Unit is meter. Default is 1m.
SpdThres(m/s)			When current ground speed is larger than this value, batching will record the location. The speed threshold is 1m/s ~ 100m/s. Unit is meter/secode. Default is 1m/s
DataFormat: (format in bitwise)			In binary package (0x01), (Default setting, The binary format please refer to Development guide)

			In NMEA package (0x02), (e.g. GGA/RMC)

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR922,60,50,10,1*00\r\n ==> Interval time: 60s, Distance threshold: 50m, Speed threshold: 10m/s, output in binary package.
Response:
$PAIR001,922,0*32\r\n ==> Success

2.3.172 Packet Type:923 PAIR_BATCHING_GET_CONFIGURATION

Get baching configuration.

DataField: $PAIR923*CS<CR><LF			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
2. $PAIR923,<IntervalThres>,<DistThres>,<SpdThres>,<DataFormat>*CS<CR><LF>
IntervalThres(s): If the time interval of recording location, batching will record the location.The interval threshold is 1s ~ 12hours. Unit is second. Default is 1s.
DistThres(m): When the distance between current and previous location exceed this value, batching will record the location. The distance threshold is 1m ~ 50000m. Unit is meter. Default is 1m.
SpdThres(m/s): When current ground speed is larger than this value, batching will record the location. The speed threshold is 1m/s ~ 100m/s. Unit is meter/secode. Default is 1m/s
DataFormat: (format in bitwise)
In binary package (0x01), (Default setting, The binary format please refer to Development guide)
In NMEA package (0x02), (e.g. GGA/RMC)
[Example]
Send:
$PAIR923*32\r\n

Response:
$PAIR001,923,0*33\r\n ==> Success
$PAIR923,60,50,10,1*01\r\n ==> Interval time: 60s, Distance threshold: 50m, Speed threshold: 10m/s, output in binary package.

2.3.173 Packet Type:924 PAIR_BATCHING_FLUSH

Flush buffer, the batching data will output as specified format.

DataField: $PAIR924*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR924*35\r\n
Response:
$PAIR001,924,0*34\r\n ==> Success

2.3.174 Packet Type:925 PAIR_BATCHING_CLEAR

Clear batching data.

DataField: $PAIR925*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]

Send:
$PAIR925*34\r\n
Response:
$PAIR001,925,0*35\r\n ==> Success

2.3.175 Packet Type:926 PAIR_BATCHING_LOG_NOW

Record the next location after this command.

DataField: $PAIR926*CS<CR><LF>			
Name	**Unit**	**Default**	**Description**
--	--	--	--

Return&Example

[Return]
1. PAIR_ACK for send result.
[Example]
Send:
$PAIR926*37\r\n
Response:
$PAIR001,926,0*36\r\n ==> Success

3 Datum List

All the datum type supported are shown in this table.

No	Datum	Region
0	WGS1984	International
1	Tokyo	Japan
2	Tokyo	Mean For Japan, South Korea, Okinawa
3	User Setting	User Setting
4	Adindan	Burkina Faso
5	Adindan	Cameroon
6	Adindan	Ethiopia
7	Adindan	Mali
8	Adindan	Mean For Ethiopia, Sudan
9	Adindan	Senegal
10	Adindan	Sudan
11	Afgooye	Somalia
12	Ain El Abd1970	Bahrain
13	Ain El Abd1970	Saudi Arabia
14	American Samoa1962	American Samoa Islands
15	Anna 1 Astro1965	Cocos Island
16	Antigua Island Astro1943	Antigua(Leeward Islands)
17	Arc1950	Botswana
18	Arc1950	Burundi
19	Arc1950	Lesotho
20	Arc1950	Malawi
21	Arc1950	Mean For Botswana, Lesotho, Malawi, Swaziland, Zaire, Zambia, Zimbabwe
22	Arc1950	Swaziland
23	Arc1950	Zaire

24	Arc1950	Zambia
25	Arc1950	Zimbabwe
26	Arc1960	Mean For Kenya Tanzania
27	Arc1960	Kenya
28	Arc1960	Tamzamia
29	Ascension Island1958	Ascension Island
30	Astro Beacon E 1945	Iwo Jima
31	Astro Dos 71/4	St Helena Island
32	Astro Tern Island (FRIG) 1961	Tern Island
33	Astronomical Station 1952	Marcus Island
34	Australian Geodetic 1966	Australia, Tasmania
35	Australian Geodetic 1984	Australia, Tasmania
36	Ayabelle Lighthouse	Djibouti
37	Bellevue (IGN)	Efate and Erromango Islands
38	Bermuda 1957	Bermuda
39	Bissau	Guuinea-Bissau
40	Bogota Observatory	Colombia
41	Bukit Rimpah	Indonesia(Bangka and Belitung Ids)
42	Camp Area Astro	Antarctica(McMurdi Camp Area)
43	Campo Inchauspe	Argentina
44	Canton Astro1966	Phoenix Island
45	Cape	South Africa
46	Cape Canaveral	Bahamas, Florida
47	Carthage	Tunisia
48	Chatham Island Astro1971	New Zealand(Chatham Island)
49	Chua Astro	Paraguay
50	Corrego Alegre	Brazil
51	Dabola	Guinea
52	Deception Island	Deception Island, Antarctia
53	Djakarta (Batavia)	Indonesia(Sumatra)
54	Dos 1968	New Georgia Islands (Gizo Island)

55	Easter Island 1967	Easter Island
56	Estonia Coordinate System1937	Estonia
57	European 1950	Cyprus
58	European 1950	Egypt
59	European 1950	England, Channel Islands, Scotland, Shetland Islands
60	European 1950	England, Ireland, Scotland, Shetland Islands
61	European 1950	Finland, Norway
62	European 1950	Greece
63	European 1950	Iran
64	European 1950	Italy (Sardinia)
65	European 1950	Italy (Slcily)
66	European 1950	Malta
67	European 1950	Mean For Austria, Belgium,Denmark, Finland, France, W Germany, Gibraltar, Greece, Italy, Luxembourg, Netherlands, Norway, Portuga,l Spain, Sweden, Switzerland
68	European 1950	Mean For Austria, Debnmark,France, W Germany, Netherland ,Switzerland
69	European 1950	Mean For Irag, Israel, Jordan, Lebanon, Kuwait, Saudi Arabia, Syria
70	European 1950	Portugal, Spain
71	European 1950	Tunisia,
72	European 1979	Mean For Austria, Finland ,Netherlands ,Norway, Spain, Sweden, Switzerland
73	Fort Thomas 1955	Nevis St Kitts (Leeward Islands)
74	Gan 1970	Republic Of Maldives
75	Geodetic Dataum 1970	New Zealand
76	Graciosa Base SW1948	Azores (Faial, Graciosa, Pico, Sao, Jorge, Terceria)
77	Guam1963	Guam
78	Gunung Segara	Indonesia (Kalimantan)
79	Gux I Astro	Guadalcanal Island
80	Herat North	Afghanistan
81	Hermannskogel Datum	Croatia-Serbia, Bosnia-Herzegoivna

82	Hjorsey 1955	Iceland
83	Hongkong 1963	Hongkong
84	Hu Tzu Shan	Taiwan
85	Indian	Bangladesh
86	Indian	India,Nepal
87	Indian	Pakistan
88	Indian 1954	Thailand
89	Indian 1960	Vietnam (Con Son Island)
90	Indian 1960	Vietnam (Near 16 deg N)
91	Indian 1975	Thailand
92	Indonesian 1974	Indonesian
93	Ireland 1965	Ireland
94	ISTS 061 Astro 1968	South Georgia Islands
95	ISTS 073 Astro 1969	Diego Garcia
96	Johnston Island 1961	Johnston Island
97	Kandawala	Sri Lanka
98	Kerguelen Island 1949	Kerguelen Island
99	Kertau 1948	West Malaysia and Singapore
100	Kusaie Astro 1951	Caroline Islands
101	Korean Geodetic System	South Korea
102	LC5 Astro 1961	Cayman Brac Island
103	Leigon	Ghana
104	Liberia 1964	Liberia
105	Luzon	Philippines (Excluding Mindanao)
106	Luzon	Philippines (Mindanao)
107	M'Poraloko	Gabon
108	Mahe 1971	Mahe Island
109	Massawa	Ethiopia (Eritrea)
110	Merchich	Morocco
111	Midway Astro 1961	Midway Islands
112	Minna	Cameroon

113	Minna	Nigeria
114	Montserrat Island Astro 1958	Montserrat (Leeward Island)
115	Nahrwan	Oman (Masirah Island)
116	Nahrwan	Saudi Arabia
117	Nahrwan	United Arab Emirates
118	Naparima BWI	Trinidad and Tobago
119	North American 1927	Alaska (Excluding Aleutian Ids)
120	North American 1927	Alaska (Aleutian Ids East of 180 degW)
121	North American 1927	Alaska (Aleutian Ids West of 180 degW)
122	North American 1927	Bahamas (Except San Salvador Islands)
123	North American 1927	Bahamas (San Salvador Islands)
124	North American 1927	Canada (Alberta, British Columbia)
125	North American 1927	Canada (Manitoba, Ontario)
126	North American 1927	Canada (New Brunswick, Newfoundland, Nova Scotia, Qubec)
127	North American 1927	Canada (Northwest Territories, Saskatchewan)
128	North American 1927	Canada (Yukon)
129	North American 1927	Canal Zone
130	North American 1927	Cuba
131	North American 1927	Greenland (Hayes Peninsula)
132	North American 1927	Mean For Antigua, Barbados, Barbuda, Caicos Islands, Cuba, Dominican, Grand Cayman, Jamaica, Turks Islands
133	North American 1927	Mean For Belize, Costa Rica, El Salvador, Guatemala, Honduras, Nicaragua
134	North American 1927	Mean For Canada
135	North American 1927	Mean For Conus
136	North American 1927	Mean For Conus (East of Mississippi, River Including Louisiana,Missouri, Minnesota)
137	North American 1927	Mean For Conus (West of Mississippi, Rive Excluding Louisiana, Minnesota, Missouri)
138	North American 1927	Mexico
139	North American 1983	Alaska (Excluding Aleutian Ids)

140	North American 1983	Aleutian Ids
141	North American 1983	Canada
142	North American 1983	Conus
143	North American 1983	Hahawii
144	North American 1983	Mexico, Central America
145	North Sahara 1959	Algeria
146	Observatorio Meteorologico 1939	Azores (Corvo and Flores Islands)
147	Old Egyptian 1907	Egypt
148	Old Hawaiian	Hawaii
149	Old Hawaiian	Kauai
150	Old Hawaiian	Maui
151	Old Hawaiian	Mean For Hawaii, Kauai, Maui, Oahu
152	Old Hawaiian	Oahu
153	Oman	Oman
154	Ordnance Survey Great Britian 1936	England
155	Ordnance Survey Great Britian 1936	England, Isle of Man, Wales
156	Ordnance Survey Great Britian 1936	Mean For England ,Isle of Man, Scotland, Shetland Island, Wales
157	Ordnance Survey Great Britian 1936	Scotland, Shetland Islands
158	Ordnance Survey Great Britian 1936	Wales
159	Pico de las Nieves	Canary Islands
160	Pitcairn Astro 1967	Pitcairn Island
161	Point 58	Mean For Burkina Faso and Niger
162	Pointe Noire 1948	Congo
163	Porto Santo 1936	Porto Santo, Maderia Islands
164	Provisional South American 1956	Bolovia
165	Provisional South American 1956	Chile (Northern Near 19 deg S)
166	Provisional South American 1956	Chile (Southern Near 43 deg S)
167	Provisional South American 1956	Colombia
168	Provisional South American 1956	Ecuador
169	Provisional South American 1956	Guyana

170	Provisional South American 1956	Mean For Bolivia Chile,Colombia, Ecuador, Guyana, Peru, Venezuela
171	Provisional South American 1956	Peru
172	Provisional South American 1956	Venezuela
173	Provisional South Chilean 1963	Chile (Near 53 deg S) (Hito XVIII)
174	Puerto Rico	Puerto Rico, Virgin Islands
175	Pulkovo 1942	Russia
176	Qatar National	Qatar
177	Qornoq	Greenland (South)
178	Reunion	Mascarene Island
179	Rome 1940	Italy (Sardinia)
180	S-42 (Pulkovo 1942)	Hungary
181	S-42 (Pulkovo 1942)	Poland
182	S-42 (Pulkovo 1942)	Czechoslavakia
183	S-42 (Pulkovo 1942)	Lativa
184	S-42 (Pulkovo 1942)	Kazakhstan
185	S-42 (Pulkovo 1942)	Albania
186	S-42 (Pulkovo 1942)	Romania
187	S-JTSK	Czechoslavakia (Prior 1 Jan1993)
188	Santo (Dos) 1965	Espirito Santo Island
189	Sao Braz	Azores (Sao Miguel, Santa Maria Ids)
190	Sapper Hill 1943	East Falkland Island
191	Schwarzeck	Namibia
192	Selvagem Grande 1938	Salvage Islands
193	Sierra Leone 1960	Sierra Leone
194	South American 1969	Argentina
195	South American 1969	Bolivia
196	South American 1969	Brazial
197	South American 1969	Chile
198	South American 1969	Colombia
199	South American 1969	Ecuador

200	South American 1969	Ecuador (Baltra, Galapagos)
201	South American 1969	Guyana
202	South American 1969	Mean For Argentina, Bolivia, Brazil,Chile, Colombia, Ecuador, Guyana, Paraguay, Peru, Trinidad and Tobago, Venezuela
203	South American 1969	Paraguay
204	South American 1969	Peru
205	South American 1969	Trinidad and Tobago
206	South American 1969	Venezuela
207	South Asia	Singapore
208	Tananarive Observatory 1925	Madagascar
209	Timbalai 1948	Brunei, E Malaysia (Sabah Sarawak)
210	Tokyo	Japan
211	Tokyo	Mean For Japan, South Korea, Okinawa
212	Tokyo	Okinawa
213	Tokyo	South Korea
214	Tristan Astro 1968	Tristam Da Cunha
215	Viti Levu 1916	Fiji (Viti Levu Island)
216	Voirol 1960	Algeria
217	Wake Island Astro 1952	Wake Atoll
218	Wake-Eniwetok 1960	Marshall Islands
219	WGS 1972	Global Definition
220	WGS 1984	Global Definition
221	Yacare	Uruguay
222	Zanderij	Suriname
223	PZ-90 v11	GLONASS