L89 R2.0&LC29H&LC79H
AGNSS Application Note

GNSS Module Series

Version: 1.0.0

Date: 2022-01-29

Status: Preliminary

At Quectel, our aim is to provide timely and comprehensive services to our customers. If you require any assistance, please contact our headquarters:

Quectel Wireless Solutions Co., Ltd.
Building 5, Shanghai Business Park Phase III (Area B), No.1016 Tianlin Road, Minhang District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local offices. For more information, please visit:
http://www.quectel.com/support/sales.htm.

For technical support, or to report documentation errors, please visit:
http://www.quectel.com/support/technical.htm.
Or email us at: support@quectel.com.

Legal Notices

We offer information as a service to you. The provided information is based on your requirements and we make every effort to ensure its quality. You agree that you are responsible for using independent analysis and evaluation in designing intended products, and we provide reference designs for illustrative purposes only. Before using any hardware, software or service guided by this document, please read this notice carefully. Even though we employ commercially reasonable efforts to provide the best possible experience, you hereby acknowledge and agree that this document and related services hereunder are provided to you on an “as available” basis. We may revise or restate this document from time to time at our sole discretion without any prior notice to you.

Use and Disclosure Restrictions

License Agreements

Documents and information provided by us shall be kept confidential, unless specific permission is granted. They shall not be accessed or used for any purpose except as expressly provided herein.

Copyright

Our and third-party products hereunder may contain copyrighted material. Such copyrighted material shall not be copied, reproduced, distributed, merged, published, translated, or modified without prior written consent. We and the third party have exclusive rights over copyrighted material. No license shall be granted or conveyed under any patents, copyrights, trademarks, or service mark rights. To avoid ambiguities, purchasing in any form cannot be deemed as granting a license other than the normal non-exclusive, royalty-free license to use the material. We reserve the right to take legal action for noncompliance with abovementioned requirements, unauthorized use, or other illegal or malicious use of the material.

Trademarks

Except as otherwise set forth herein, nothing in this document shall be construed as conferring any rights to use any trademark, trade name or name, abbreviation, or counterfeit product thereof owned by Quectel or any third party in advertising, publicity, or other aspects.

Third-Party Rights

This document may refer to hardware, software and/or documentation owned by one or more third parties ("third-party materials"). Use of such third-party materials shall be governed by all restrictions and obligations applicable thereto.

We make no warranty or representation, either express or implied, regarding the third-party materials, including but not limited to any implied or statutory, warranties of merchantability or fitness for a particular purpose, quiet enjoyment, system integration, information accuracy, and non-infringement of any third-party intellectual property rights with regard to the licensed technology or use thereof. Nothing herein constitutes a representation or warranty by us to either develop, enhance, modify, distribute, market, sell, offer for sale, or otherwise maintain production of any our products or any other hardware, software, device, tool, information, or product. We moreover disclaim any and all warranties arising from the course of dealing or usage of trade.

Privacy Policy

To implement module functionality, certain device data are uploaded to Quectel's or third-party's servers, including carriers, chipset suppliers or customer-designated servers. Quectel, strictly abiding by the relevant laws and regulations, shall retain, use, disclose or otherwise process relevant data for the purpose of performing the service only or as permitted by applicable laws. Before data interaction with third parties, please be informed of their privacy and data security policy.

Disclaimer

a) We acknowledge no liability for any injury or damage arising from the reliance upon the information.
b) We shall bear no liability resulting from any inaccuracies or omissions, or from the use of the information contained herein.
c) While we have made every effort to ensure that the functions and features under development are free from errors, it is possible that they could contain errors, inaccuracies, and omissions. Unless otherwise provided by valid agreement, we make no warranties of any kind, either implied or express, and exclude all liability for any loss or damage suffered in connection with the use of features and functions under development, to the maximum extent permitted by law, regardless of whether such loss or damage may have been foreseeable.
d) We are not responsible for the accessibility, safety, accuracy, availability, legality, or completeness of information, advertising, commercial offers, products, services, and materials on third-party websites and third-party resources.

About the Document

Document Information	
Title	**L89 R2.0&LC29H&LC79H AGNSS Application Note**
Subtitle	GNSS Module Series
Document Type	Application Note
Document Status	Preliminary

Revision History

Revision	Date	Description
-	2021-10-14	Creation of the document
1.0.0	2022-01-29	Preliminary

Contents

About the Document **3**
Contents **4**
Table Index **6**
Figure Index **7**

1 Introduction **8**
1.1. Differences Between Host EPO and Flash EPO 8
1.2. AGNSS Requirements 9

2 Download of EPO Files **10**
2.1. Get EPO Files from Server 10
2.2. EPO Files Format 11
2.2.1. EPO Files Format – GPS Only 12
2.2.2. EPO File Format – BDS/Galileo Only 13
2.2.3. EPO File Format – GPS + GLONASS 15
2.3. Types of EPO Files 15
2.4. Recommended Download Procedures of EPO Files 16
2.5. The Validity Period of EPO Files 17

3 AGNSS Implementation **19**
3.1. AGNSS with Flash EPO 19
3.1.1. Binary Protocol 19
3.1.2. EPO Data Transfer Protocol 21
3.1.3. AGNSS Procedure with Flash EPO 25
3.2. AGNSS with Host EPO 26
3.2.1. Recommended Sequence for Host EPO 26
3.2.2. Sample Code to Send EPO 27

4 AGNSS Related Messages **30**
4.1. PAIR001 PAIR_ACK 30
4.2. PAIR010 PAIR_REQUEST_AIDING 30
4.3. PAIR470 PAIR_EPO_GET_STATUS 32
4.4. PAIR471 PAIR_EPO_SET_DATA 33
4.5. PAIR472 PAIR_EPO_ERASE_FLASH_DATA 34
4.6. PAIR590 PAIR_TIME_SET_REF_UTC 35
4.7. PAIR600 PAIR_LOC_SET_REF 35

5 EPO Usage Through QGNSS **37**
5.1. Download Flash EPO with QGNSS 37
5.2. Download Host EPO with QGNSS 38

6 AGNSS Implementation Example **40**
6.1. Flash EPO Implementation 40
6.2. Host EPO Implementation 41

7 Appendix References 43

Table Index

Table 1: Differences between Flash EPO and Host EPO 8
Table 2: AGNSS Related Commands 9
Table 3: Download URL of EPO Files 10
Table 4: EPO SVID Range 11
Table 5: Types of EPO Files 15
Table 6: Description of Binary Protocol Fields 20
Table 7: Start of EPO Binary Format 20
Table 8: EPO Data Binary Format 20
Table 9: End of EPO Binary Format 20
Table 10: Terms and Abbreviations 43

Figure Index

Figure 1: EPO Files Format – GPS Only 12
Figure 2: Format for Several Segments of EPO Files 12
Figure 3: EPO Files Format – BDS/Galileo Only 13
Figure 4: Galileo EPO Header 14
Figure 5: BDS EPO Header 14
Figure 6: EPO Files Format – GPS + GLONASS 15
Figure 7: Recommended Download Procedures of EPO Files 16
Figure 8: Structure of Binary Protocol 19
Figure 9: AGNSS Procedure with Flash EPO 25
Figure 10: Suggested Sequence for Host EPO 27
Figure 11: Flash EPO Setting Interface of QGNSS 37
Figure 12: EPO File Downloading 38
Figure 13: Host EPO Setting Interface of QGNSS 39
Figure 14: Host EPO File Downloading 39

1 Introduction

EPO (Extended Prediction Orbit) is an AGNSS feature provided by the chipset supplier, which enables the receiver to minimize TTFF and improve accuracy in weak signal conditions. This document mainly describes the download of EPO files, AGNSS implementation, EPO related PAIR commands and how to evaluate the EPO functionality through QGNSS tool.

1.1. Differences Between Host EPO and Flash EPO

Both Flash EPO and Host EPO allow the GNSS receiver to achieve a shorter TTFF, but their differences make each of them suitable for different applications.

Host EPO (also called Real Time AGNSS) allows the receiver to store in RAM up to 6 hours of assistance data which are sent to the receiver through NMEA PAIR commands listed in ***Chapter 4***. For Host EPO, there is no data retention after the GNSS receiver reboots and the data should be re-downloaded.

Flash EPO, on the other hand, allows the receiver to store in Flash 7 or 14 days' assistance data which are sent to the receiver through Binary Protocol defined by the chipset supplier. Flash EPO enables the receiver to reuse all assistance information stored in flash before the information expires. See ***Chapter 2.5*** for the validity period of EPO files.

Table 1: Differences between Flash EPO and Host EPO

Type	Flash EPO	Host EPO
Storage space	Flash	RAM
Storage capacity	7 or 14 days' assistance data	6 hours' assistance data
Protocol	Binary	NMEA

NOTE

The maximum period that EPO data can be stored in Flash is 14 days for GPS-only EPO files and GPS + GLONASS EPO files, 7 days for Galileo EPO files and 3 days for BDS EPO files. If a 30-day GPS-only EPO file and GPS + GLONASS is sent, only the first 14 days of EPO data will be stored.

1.2. AGNSS Requirements

The host needs to provide the Reference Time, Reference Position and EPO data to the GNSS receiver. The information provided by the host needs to meet the following requirements so that the GNSS receiver can make better use of EPO:

- The **Reference Time** should be accurate within 3 s and must be specified in UTC time.
- The **Reference Position** should be accurate within 30 km around the true position of the receiver. Keep in mind that if the receiver's view of the sky is limited, the accuracy of the Reference Position needs to be increased.
- The **EPO data** should be valid.

The receiver can benefit from any of the assistance data to improve the TTFF. All assistance data (Reference Time, Reference Position and EPO data) are useful but none of them are mandatory. If some of them are not available or have expired, it is recommended to avoid using them.

The host can send the Reference Time, Reference Position and EPO data to the GNSS receiver through the messages listed in following table. See ***Chapter 4*** for a detailed description of these messages.

Table 2: AGNSS Related Commands

Packet Type	Data Content
PAIR471	GPS/GLONASS/Galileo/BDS EPO data for a single satellite.
PAIR590	Reference UTC Time.
PAIR600	Reference Location.

2 Download of EPO Files

Quectel does not provide any Service Level Agreement for EPO files. It is recommended to download EPO data to your own server and send them to devices so as to ensure the availability of EPO data.

2.1. Get EPO Files from Server

Table 3: Download URL of EPO Files

EPO Type	GNSS Type	EPO File URL	File Name
Unified QEPO	GPS only	http://wpepodownload.mediatek.com/QGPS.DAT?*vendorinfo*	Single name: QGPS.DAT
Unified QEPO	GPS + GLONASS	http://wpepodownload.mediatek.com/QG_R.DAT?*vendorinfo*	Single name: QG_R.DAT
Unified QEPO	BDS only	http://wpepodownload.mediatek.com/QBD2.DAT?vendorinfo	Single name: QBD2.DAT
Unified QEPO	Galileo only	http://wpepodownload.mediatek.com/QGA.DAT?vendorinfo	Single name: QGA.DAT
EPO	GPS only	http://wpepodownload.mediatek.com/EPO_GPS_3_***X***.DAT?*vendorinfo*	**X** = 1~10 EPO_GPS_3_1.DAT to EPO_GPS_3_10.DAT
EPO	GPS + GLONASS	http://wpepodownload.mediatek.com/EPO_GR_3_***X***.DAT?*vendorinfo*	**X** = 1~10. EPO_GR_3_1.DAT to EPO_GR_3_10.DAT
EPO	BDS only	http://wpepodownload.mediatek.com/EPO_BDS_3.DAT?vendorinfo	EPO_BDS_3.DAT
EPO	Galileo only	http://wpepodownload.mediatek.com/EPO_GAL_***X***.DAT?vendorinfo	**X** = 3 or 7 EPO_GAL_3.DAT or EPO_GAL_7.DAT

The following shows a complete URL sample:
http://wpepodownload.mediatek.com/QGPS.DAT?*vendor=AAA&project=BBB&device_id=CCC*

- The query string starts with "*?*" and is separated by "*&*".
- The values of "*vendor*" and "*project*" (AAA, BBB in the example) are issued by Quectel, contact

Quectel Technical Supports to get the value.
The value of "*device_id*" (CCC in the example) contains two parts – one is assigned by Quectel and the other assigned by the customer. For example: if *CCC* = *XXX_YYY*, the value *XXX* is provided by Quectel and you can contact Quectel Technical Supports to get the value, while *YYY* can be assigned by yourself and it must be a unique value, such as IMEI. Each device must have a unique ID.

NOTE

As there can be a maximum of 30 days' predictions, there will be up to 10 files.
Slices of 30-day EPO:
_1 for days 1 to 3,
_2 for days 4 to 6,
...
_10 for days 28 to 30.

2.2. EPO Files Format

This part mainly illustrates the format of EPO files.
The SV ID of EPO file for different constellations can be referred to in *Table 4: EPO SVID Range*.

Table 4: EPO SVID Range

GNSS Type	PRN	EPO SVID
GPS	1~32	1~32
GLONASS	1~24	65~88
Galileo	1~36	101~136
BDS	1~54, 55~63	201~254, 190~198

2.2.1. EPO Files Format – GPS Only

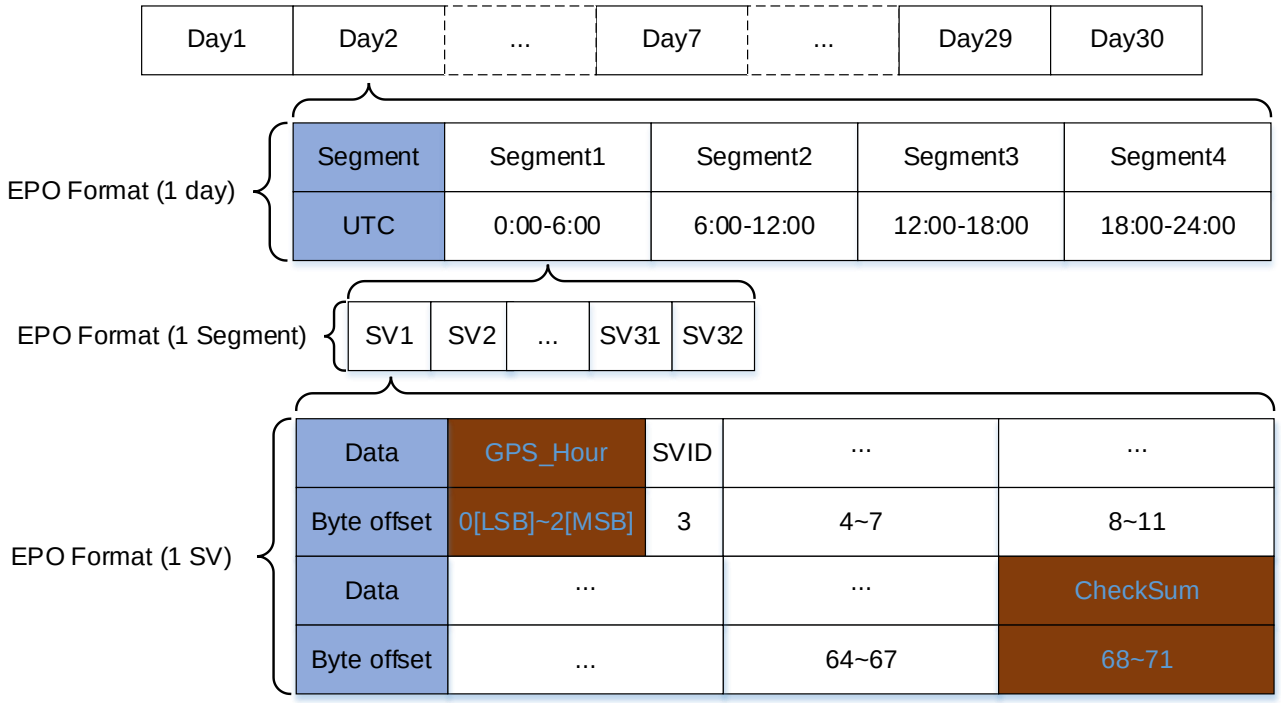

Figure 1: EPO Files Format – GPS Only

GPS_Secs = GPS_Hour * 3600
GPS_Week Number = GPS_Secs / 604800
GPS TOW = GPS_Secs % 604800

An EPO file contain GPS Time (GPS_Week, GPS_Hour and GPS_Secs). The maximum unit in GPS Time is GPS week which starts at approximately midnight of January 5th to 6th, 1980.

The following figure illustrates the format for several segments of EPO files.

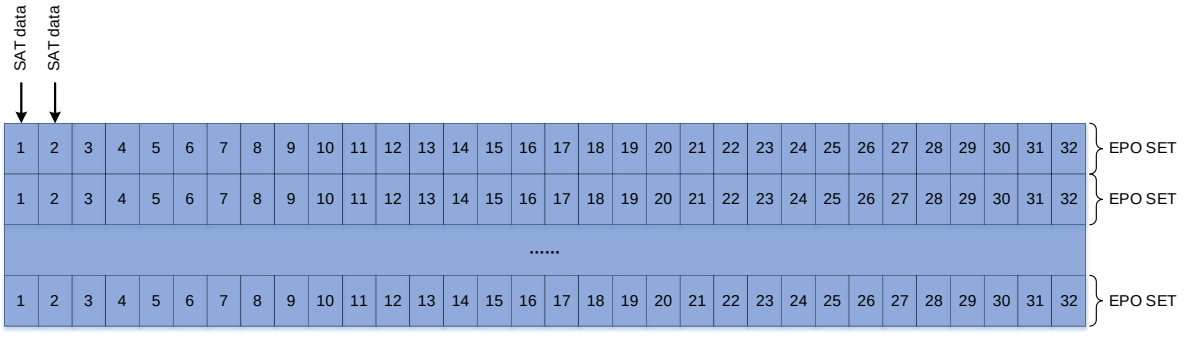

Figure 2: Format for Several Segments of EPO Files

The basic unit of an EPO file is SAT Data and the size of each SAT Data is 72 bytes. One EPO SET contains 32 SAT Data, so the data size of an EPO SET is 2304 bytes. Each EPO file contains several EPO SETs so the file size must be a multiple of 2304 bytes. An EPO SET is valid for 6 hours. Therefore, there will be 4 EPO SETs for one day.

2.2.2. EPO File Format – BDS/Galileo Only

Galileo EPO data consist of the 72-byte header and 3 or 7 days of fundamental EPO data. BDS EPO data consist of the 72-byte header and 3 days of EPO data only. The EPO format for both has no fixed size.

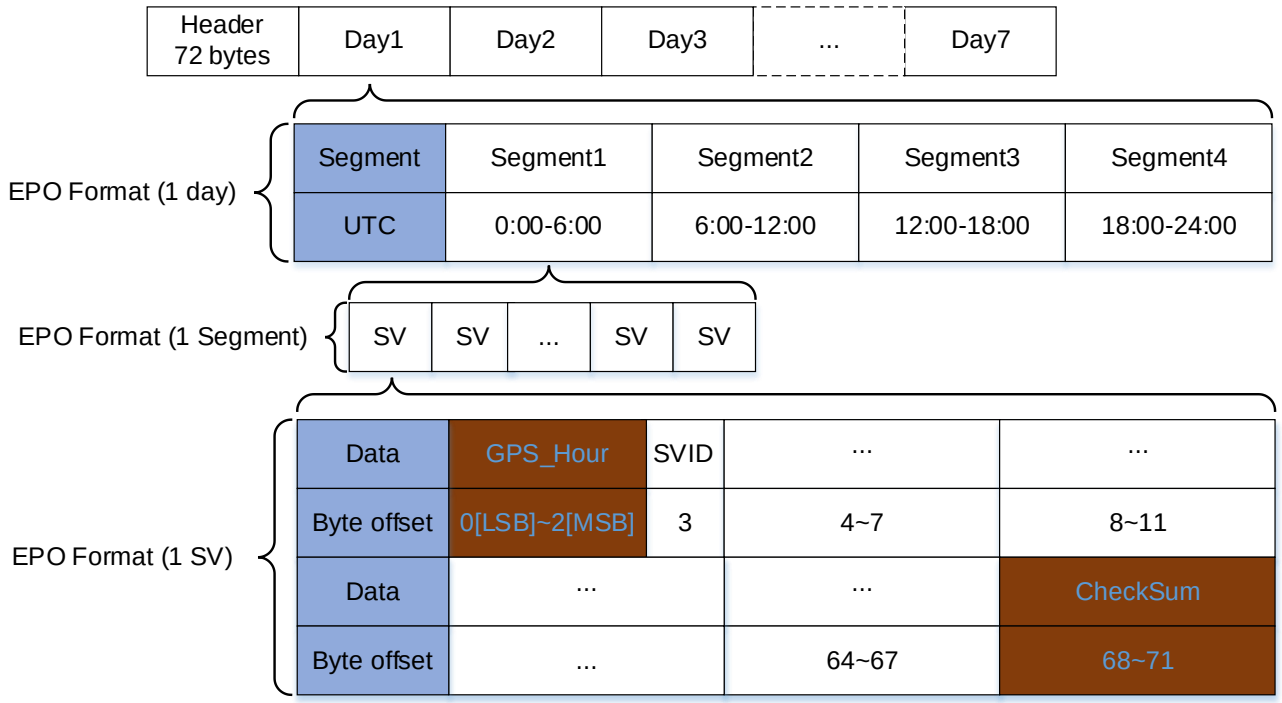

Figure 3: EPO Files Format – BDS/Galileo Only

The 72-byte header contains the SV available bitmask, and calculate the available satellite ID. When the SV available bit mask position is 1, it indicates that the satellite is available, and the number of bits indicates the satellite ID.
For example, you can parse the data from *Figure 4: Galileo EPO Header*.

- SVID is FE for Galileo.
- SV available bitmask: 09 67 94 5D DF.
- Total available SV is 22.
- Available SV: 1, 2, 3, 4, 5, 7, 8, 9, 11, 12, 13, 15, 19, 21, 24, 25, 26, 27, 30, 31, 33, 36.

Data	***	SVID	SV available bitmask(Lbytes)	***	***
Byte offset	0~2	3	4[LSB]~7[MSB]	8~11	12~15
example		FE	DF 5D 94 67		

Data	SV available bitmask(Hbytes)	***	***	***
Byte offset	16[LSB]~19[MSB]	20~23	24~27	18~31
example	09 00 00 00			

Figure 4: Galileo EPO Header

For example, you can parse the data from *Figure 5: BDS EPO Header.*

- SVID is FF for BDS.
- SV available bitmask: 3F FF BF FC 3F FF.
- Total available SV is 41.
- Available SV: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46.

Data	***	SVID	SV available bitmask(Lbytes)	***	***
Byte offset	0~2	3	4[LSB]~7[MSB]	8~11	12~15
example		FF	FF 3F FC BF		

Data	SV available bitmask(Hbytes)	***	***	***
Byte offset	16[LSB]~19[MSB]	20~23	24~27	18~31
example	FF 3F 00 00			

Figure 5: BDS EPO Header

2.2.3. EPO File Format – GPS + GLONASS

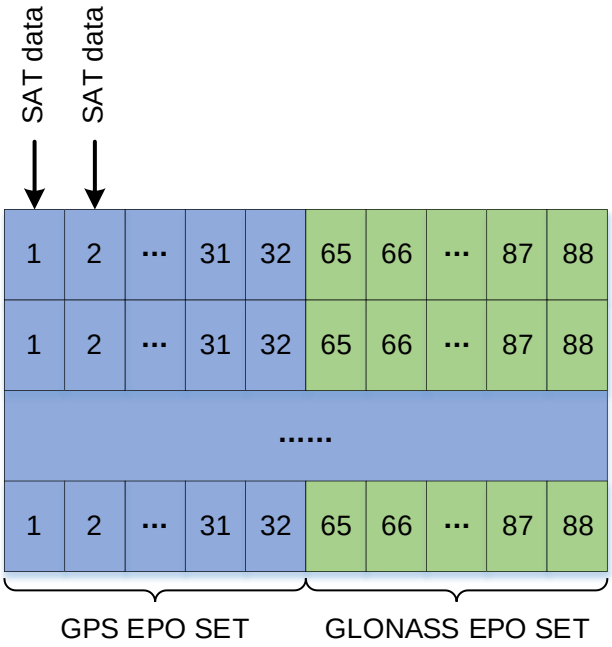

Figure 6: EPO Files Format – GPS + GLONASS

The basic unit of an EPO file is SAT Data, and the size of a SAT Data is 72 bytes. In GPS + GLONASS EPO files, one EPO SET contains 56 SAT Data, so the data size for an EPO SET is 4032 bytes. Each EPO file contains several EPO SETs. The file size must be a multiple of 4032 bytes. An EPO SET is valid for 6 hours. Therefore, there will be 4 EPO SETs for one day.

2.3. Types of EPO Files

The EPO data can be downloaded in the form of files. You can select the most suitable file to download based on the availability of a data connection and storage space of your application. See *Table 3: Download URL of EPO Files* and *Table 5: Types of EPO Files* to decide on the file type to be downloaded.

Table 5: Types of EPO Files

EPO Type	GNSS Type	Description
EPO	GPS only	3-14 days prediction orbit (ephemeris). Split in 5 files containing 3 days of information each.
EPO	Galileo only	3/7-day prediction orbit (ephemeris).

EPO	BDS only	3-day prediction orbit (ephemeris).
EPO	GPS + GLONASS	3-14 days prediction orbit (ephemeris). Split in 5 files containing 3 days of information each.
Unified QEPO	GPS only	6-hour prediction orbit (ephemeris). Single file containing the latest GPS EPO data available.
Unified QEPO	GPS + GLONASS	6-hour prediction orbit (ephemeris). Single file, containing the latest GPS + GLONASS EPO data available.
Unified QEPO	BDS/Galileo only	6-hour prediction orbit (ephemeris). Single file, containing the latest BDS/Galileo EPO data available.

2.4. Recommended Download Procedures of EPO Files

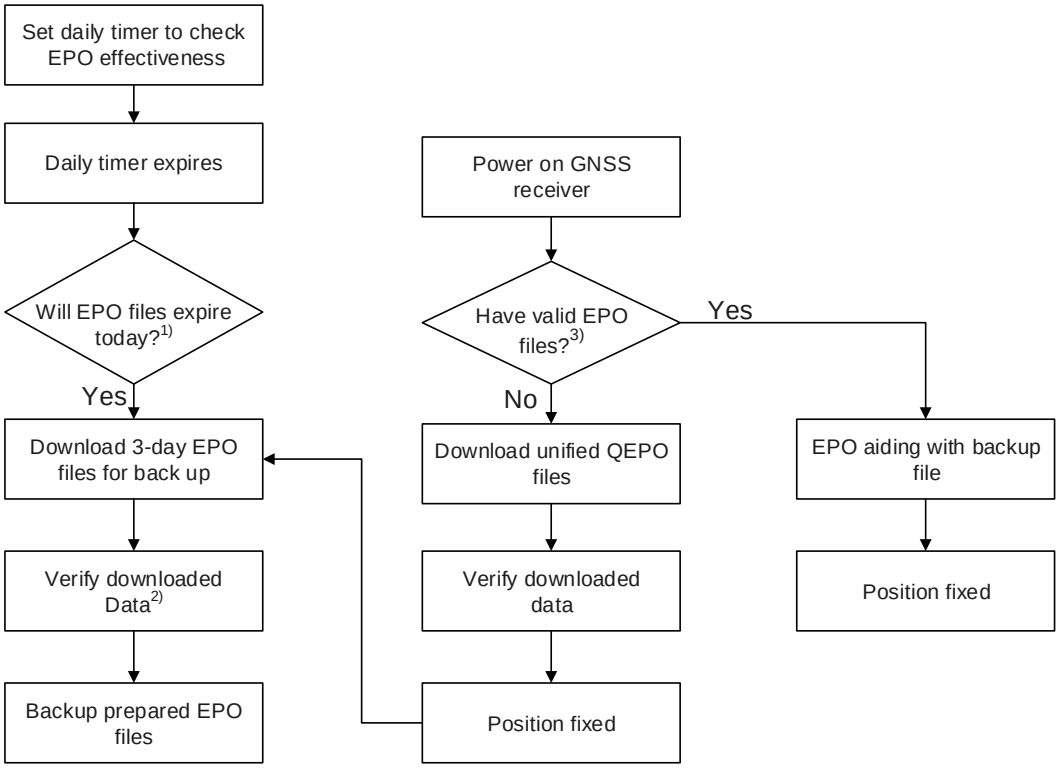

Figure 7: Recommended Download Procedures of EPO Files

NOTE

1. Users must know the current UTC time so as to download the current valid EPO files.
2. Send **PAIR470** for checking whether the data are correct.
3. If the device is powered off for a long time, EPO files stored in flash will be expired.

2.5. The Validity Period of EPO Files

EPO validity period is related to the current UTC time. The EPO validity period can be obtained from the last segment of the EPO file. See *Figure 1: EPO Files Format – GPS Only* or the sample of how to calculate EPO validity period (GPS_Hour + 6). It is necessary to download the EPO file 12 hours in advance. The following codes show the conversion between UTC time and GPS time.

```
void utc_to_gpstime(kal_uint32  year,        //Input year
                    kal_uint8   mon,         //Input month: 1~12
                    kal_uint8   day,         //Input day: 1~31
                    kal_uint8   hour,        //Input hour: 0~23
                    kal_uint8   min,         //Input Minute: 0~59
                    kal_uint8   sec,         //Input second: 0~59
                    kal_int32*  wn,          //Output GPS week number
                    double*    tow)          //Output GPS time of week
{
    kal_int32 iYearsElapsed;                 //Elapsed years since 1980
    kal_int32 iDaysElapsed;                  //Elapsed days since Jan 5/Jan 6, 1980
    kal_int32 iLeapDays;                     //Leap days since Jan 5/Jan 6, 1980
    kal_int32 i;
    //Number of days at the start of each month (ignore leap years).
    kal_uint16 doy[12] = {0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334};

    iYearsElapsed = year - 1980;
    i = 0;
    iLeapDays = 0;
    while (i <= iYearsElapsed)
    {
        if ((i % 100) == 20)
        {
            if ((i % 400) == 20)
            {
                iLeapDays++;
            }
        }
        else if ((i % 4) == 0)
        {
            iLeapDays++;
        }
        i++;
    }
    /*  iLeapDays = iYearsElapsed / 4 + 1; */.
    if ((iYearsElapsed % 100) == 20)
    {
        if (((iYearsElapsed % 400) == 20) && (mon <= 2))
        {
            iLeapDays--;
        }
    }
    else if (((iYearsElapsed % 4) == 0) && (mon <= 2))
    {
        iLeapDays--;
    }
```

```
    iDaysElapsed = iYearsElapsed * 365 + doy[mon - 1] + day + iLeapDays - 6;
    //Convert time to GPS weeks and seconds.
    *wn = iDaysElapsed / 7;
    *tow = (double)(iDaysElapsed % 7) * 86400 + hour * 3600 + min * 60 + sec;
}
```

3 AGNSS Implementation

This chapter describes two AGNSS implementation methods: Host EPO and Flash EPO.

- Implement AGNSS with Host EPO

The host sends EPO data to the GNSS receiver through NMEA PAIR command, such as **PAIR471**.

- Implement AGNSS with Flash EPO

The EPO data are downloaded to the flash of GNSS receiver through Binary Protocol.

Flash EPO keeps data for a longer time than Host EPO.

3.1. AGNSS with Flash EPO

Flash EPO can store up to 14 days of EPO assistance data on flash, which enables the receiver to make use of the available data since boot time. The communication protocol of Flash EPO is Binary Protocol. In order to download assistance data to the receiver, the user need to download assistance data in the binary format specified in this document. See ***Chapter 3.1.2*** and ***Chapter 3.1.3*** for details.

3.1.1. Binary Protocol

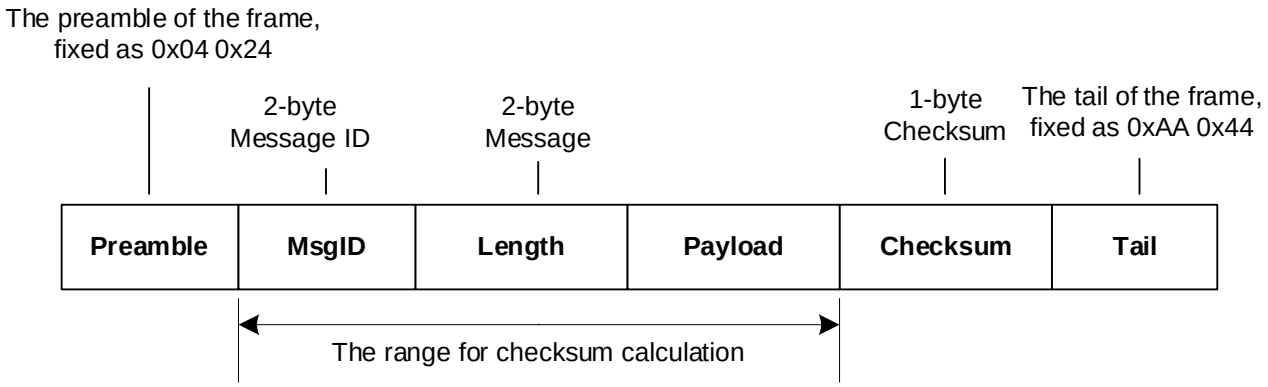

Figure 8: Structure of Binary Protocol

Table 6: Description of Binary Protocol Fields

Field	Length (Byte)	Description
Preamble	2	Fixed as 0x2404. Use little endian.
MsgID	2	Message ID.
Length	2	The length of the **Payload**. Unit: byte. The packet size is 72 bytes by default. Use little endian.
Payload	Variable	Payload data to be transferred.
Checksum	1	The checksum is the 8-bit exclusive OR of all bytes in the message between (but not including) the **Preamble** and the **Checksum**.
Tail	2	Fixed as 0x44AA. Use little endian.

The EPO binary format is divided into start message, EPO data message, and end message.

Table 7: Start of EPO Binary Format

Preamble	MsgID	Length	Payload	Checksum	Tail
0x04 0x24	0xB0 0x04	1	'G' – GPS 'R' – GLONASS 'E' – Galileo 'C' – BDS	0x**	0xAA 0x44
2 Bytes	2 Bytes	2 Bytes	2 Bytes	1 Byte	2 Bytes

Table 8: EPO Data Binary Format

Preamble	MsgID	Length	Payload	Checksum	Tail
0x04 0x24	0xB1 0x04	72	EPO Data	0x**	0xAA 0x44
2 Bytes	2 Bytes	2 Bytes	72 Bytes	1 Byte	2 Bytes

Table 9: End of EPO Binary Format

Preamble	MsgID	Length	Payload	Checksum	Tail
0x04 0x24	0xB2 0x04	1	'G' – GPS 'R' – GLONASS	0x**	0xAA 0x44

			‘E’ – Galileo ‘C’ – BDS		
2 Bytes	2 Bytes	2 Bytes	2 Bytes	1 Byte	2 Bytes

3.1.2. EPO Data Transfer Protocol

The sequence of EPO transmission is that the host sends the start packet, then splits the EPO data into the data packet and sends it, and finally sends the end packet. The host should follow the EPO Data Transfer Protocol when transferring EPO data to GNSS receiver.

3.1.2.1 Pseudo Code for EPO Data Transfer Protocol

The following shows pseudo code for the EPO data transfer procedure of GPS + GLONASS, which are for reference only.

```
#define GNSS_APP_BINARY_BINARY_PREAMBLE1      (0x04)
#define GNSS_APP_BINARY_BINARY_PREAMBLE2      (0x24)
#define GNSS_APP_BINARY_BINARY_ENDWORD1       (0xAA)
#define GNSS_APP_BINARY_BINARY_ENDWORD2       (0x44)

#define GNSS_APP_BINARY_BINARY_PREAMBLE_SIZE   (2)
#define GNSS_APP_BINARY_BINARY_CHECKSUM_SIZE   (1)
#define GNSS_APP_BINARY_BINARY_ENDWORD_SIZE    (2)
#define GNSS_APP_BINARY_BINARY_CONTROL_SIZE
       (GNSS_APP_BINARY_BINARY_PREAMBLE_SIZE + \
       GNSS_APP_BINARY_BINARY_CHECKSUM_SIZE + \
       GNSS_APP_BINARY_BINARY_ENDWORD_SIZE)

#define GNSS_APP_BINARY_BINARY_MESSAGE_ID_SIZE      (2)
#define GNSS_APP_BINARY_BINARY_PAYLOAD_LENGTH_SIZE   (2)
#define GNSS_APP_BINARY_BINARY_PAYLOAD_HEADER_SIZE
       (GNSS_APP_BINARY_BINARY_MESSAGE_ID_SIZE + \
       GNSS_APP_BINARY_BINARY_PAYLOAD_LENGTH_SIZE)

#define GNSS_APP_BINARY_BINARY_MAX_DATA_SIZE (512)
#define GNSS_APP_BINARY_BINARY_MAX_PAYLOAD_DATA_SIZE
       (GNSS_APP_BINARY_BINARY_MAX_DATA_SIZE - \
       GNSS_APP_BINARY_BINARY_CONTROL_SIZE - \
       GNSS_APP_BINARY_BINARY_PAYLOAD_HEADER_SIZE)

typedef enum gnss_app_binary_binary_decode_results {
    GNSS_APP_BINARY_BINARY_DECODE_SUCCESS = 0,
```

```
    GNSS_APP_BINARY_BINARY_DECODE_WRONG_PARAMETER = -1,
    GNSS_APP_BINARY_BINARY_DECODE_WRONG_PREAMBLE  = -2,
    GNSS_APP_BINARY_BINARY_DECODE_WRONG_CHECKSUM  = -3,
    GNSS_APP_BINARY_BINARY_DECODE_WRONG_ENDWORD   = -4,
}gnss_app_binary_binary_decode_results_t;
typedef struct gnss_app_binary_binary_payload {
    uint16_t message_id;
    uint16_t data_size; /* actual size of data in payload data buffer */
    uint8_t data[GNSS_APP_BINARY_BINARY_MAX_PAYLOAD_DATA_SIZE];
}gnss_app_binary_binary_payload_t;

uint8_t gnss_app_binary_calculate_binary_checksum(const
    gnss_app_binary_binary_payload_t* const payload)
{
    uint8_t checksum = 0;
    uint8_t* pheader = NULL;
    uint8_t* pdata = NULL;
    uint16_t i;
    if (NULL == payload) {
        return 0;
    }
    /* The checksum is the 8-bit exclusive OR of all bytes in the payload. */
    pheader = (uint8_t*)payload;
    for (i = 0; i < GNSS_APP_BINARY_BINARY_PAYLOAD_HEADER_SIZE; i++) {
        checksum ^= *pheader;
        pheader++;
    }
    pdata = (uint8_t*)payload->data;
    for (i = 0; i < payload->data_size; i++) {
        checksum ^= *pdata;
        pdata++;
    }
    return checksum;
}

int16_t gnss_app_binary_encode_binary_packet(uint8_t* const buffer, uint16_t
    max_buffer_size, const gnss_app_binary_binary_payload_t* const payload)
{
    uint8_t* pbyte;
    uint16_t required_length;
    if (NULL == buffer || payload == NULL) {
        return -1;
    }
    required_length = payload->data_size + GNSS_APP_BINARY_BINARY_CONTROL_SIZE +
```

```
        GNSS_APP_BINARY_BINARY_PAYLOAD_HEADER_SIZE;
    if (max_buffer_size < required_length) {
        return -1;
    }
    memset((void*)buffer, 0, max_buffer_size);
    buffer[0] = GNSS_APP_BINARY_BINARY_PREAMBLE1;
    buffer[1] = GNSS_APP_BINARY_BINARY_PREAMBLE2;
    pbyte = &buffer[2];
    memcpy(pbyte, payload, GNSS_APP_BINARY_BINARY_PAYLOAD_HEADER_SIZE);
    pbyte += GNSS_APP_BINARY_BINARY_PAYLOAD_HEADER_SIZE;
    memcpy(pbyte, payload->data, payload->data_size);
    pbyte += payload->data_size;
    *pbyte++ = gnss_app_binary_calculate_binary_checksum(payload);
    *pbyte++ = GNSS_APP_BINARY_BINARY_ENDWORD1;
    *pbyte = GNSS_APP_BINARY_BINARY_ENDWORD2;
    return required_length;
}

FILE* gnss_epo_file = NULL;
#define GNSS_MAX_EPO_NUMBER (37)
#define GNSS_MAX_RECORD_SIZE (72)
static uint32_t gnss_epo_sv_buf[(GNSS_MAX_EPO_NUMBER *
    GNSS_MAX_RECORD_SIZE) / sizeof(uint32_t)];
int16_t gnss_epo_encode_binary(uint16_t msg_id, char* buffer, uint16_t buffer_size,
    char* data_input, int32_t data_length) {
    gnss_app_binary_binary_payload_t payload;
    int16_t binary_message_length;
    memset((void*)&payload, 0, sizeof(gnss_app_binary_binary_payload_t));
    payload.message_id = msg_id;
    payload.data_size = (uint16_t)data_length;
    memcpy(payload.data, data_input, sizeof(uint8_t) * data_length);
    binary_message_length = gnss_app_binary_encode_binary_packet(buffer,
        buffer_size, &payload);
    return binary_message_length;
}

void gnss_epo_binary_demo() {
    gnss_app_binary_data_t data;
    gnss_app_binary_data_result_t result = { 0 };
    uint32_t* epobuf;
    int32_t i;
    char buffer[500];
    uint16_t length = 0;
    int32_t buffer_size = 0;
```

```
    uint8_t segment = 0;

    uint8_t curr_sys_type = 'G';  //type is the GPS
    gnss_epo_file = fopen("EPO_GR_3_1.DAT", "rb");
    length = gnss_epo_encode_binary(1200, buffer, 512, &curr_sys_type, 1);
    gnss_app_uart_send_data(buffer, length);
    memset(&gnss_epo_sv_buf, 0, sizeof(gnss_epo_sv_buf));
    while (gnss_epo_read_data(&gnss_epo_sv_buf, 32 * GNSS_MAX_RECORD_SIZE, segment *
        (32 + 24) * GNSS_MAX_RECORD_SIZE)) {
        segment++;
        for (i = 0; i < 32; i++) {
            epobuf = (uint32_t*)(gnss_epo_sv_buf + ((i * GNSS_MAX_RECORD_SIZE) /
                4));
            length = gnss_epo_encode_binary(1201, buffer, 512, (char*)epobuf,
                GNSS_MAX_RECORD_SIZE);
            gnss_app_uart_send_data(buffer, length);
        }
        memset(&gnss_epo_sv_buf, 0, sizeof(gnss_epo_sv_buf));
    }
    length = gnss_epo_encode_binary(1202, buffer, 512, &curr_sys_type, 1);
    gnss_app_uart_send_data(buffer, length);

    curr_sys_type = 'R';  //type is the GLONASS
    length = gnss_epo_encode_binary(1200, buffer, 512, &curr_sys_type, 1);
    gnss_app_uart_send_data(&data, &result);
    memset(&gnss_epo_sv_buf, 0, sizeof(gnss_epo_sv_buf));
    segment = 0;
    while (gnss_epo_read_data(&gnss_epo_sv_buf, 37 * GNSS_MAX_RECORD_SIZE, (segment
        * (32 + 24) * GNSS_MAX_RECORD_SIZE) + (32 * GNSS_MAX_RECORD_SIZE))) {
        segment++;
        for (i = 0; i < 24; i++) {
            epobuf = (uint32_t*)(gnss_epo_sv_buf + ((i * GNSS_MAX_RECORD_SIZE) /
                4));
            length = gnss_epo_encode_binary(1201, buffer, 512, (char*)epobuf,
                GNSS_MAX_RECORD_SIZE);
            gnss_app_uart_send_data(buffer, length);
        }
        memset(&gnss_epo_sv_buf, 0, sizeof(gnss_epo_sv_buf));
    }
    length = gnss_epo_encode_binary(1202, buffer, 512, &curr_sys_type, 1);
    gnss_app_uart_send_data(buffer, length);
    fclose(gnss_epo_file);
}
```

3.1.3. AGNSS Procedure with Flash EPO

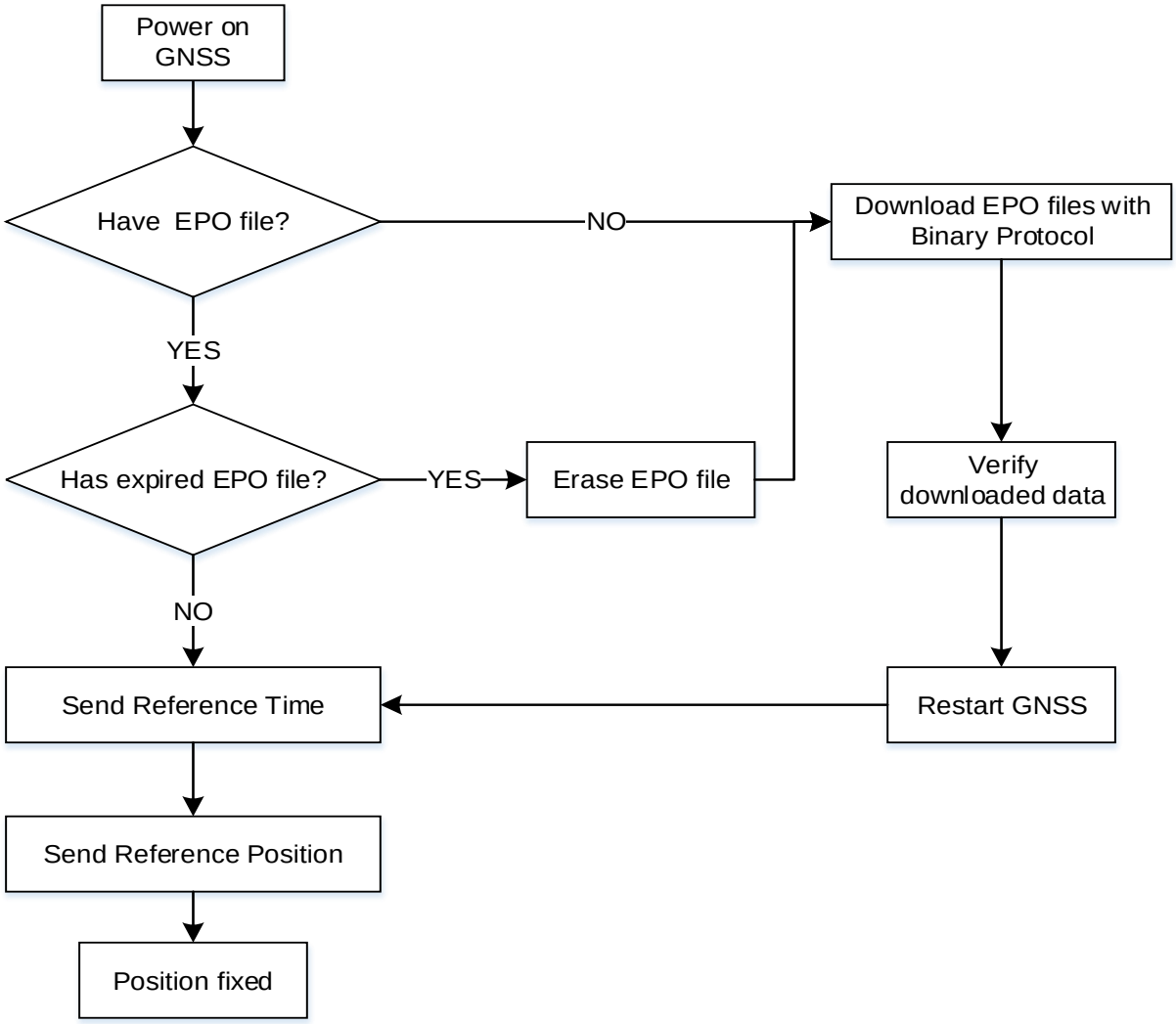

Figure 9: AGNSS Procedure with Flash EPO

1. Power on the GNSS module.
2. Check whether there are EPO data in GNSS module through **PAIR470**.
3. If there exists EPO data, go to the next step to check the data validity. Otherwise, download EPO data to GNSS module and verify downloaded data, then restart GNSS module and directly go to step 6.
4. Check whether the EPO file in GNSS module has expired.
5. If the EPO data is not expired, please go to next step. Otherwise erase expired EPO file with **PAIR472** and download a new EPO file.
6. Send reference time to GNSS module with **PAIR590**.
7. Send reference location to GNSS module with **PAIR600**.
8. Wait for the GNSS module to fix position.

3.2. AGNSS with Host EPO

Host EPO allows for a simpler text-based implementation which enables the receiver to perform a fast start up where assistance data must be sent to the receiver each time it boots. When using Host EPO, the receiver can only receive one block of assistance data valid for 6 hours.

Implementing Host EPO only requires a few PAIR sentences and all data transfer can be done in NMEA mode. See ***chapter 4*** for description of **PAIR471**, **PAIR590** and **PAIR600**.

3.2.1. Recommended Sequence for Host EPO

After the GNSS receiver is powered on, it will output startup messages **$PAIR010,1,-1*16** and **$PAIR010,2,-1*15** to notify that the expiration of GNSS aiding data stored in the module. After the host receives the system startup message, it can send the assistance data in sequence shown in the *Figure 10: Suggested Sequence for Host EPO*. The sequence of assistance data is Reference Time, Reference Position and EPO data.

The Host EPO procedure consists of the following steps:

1. GNSS module starts up;
2. The host sends Reference Time;
3. The host sends Reference Position;
4. The host sends EPO data.

The supplied Reference Time, Reference Position and EPO data must comply with the requirements listed in ***Chapter 1.2***.

NOTE

In the current implementation, the host needs to wait for a **PAIR001** packet to be returned before sending another segment of EPO data.

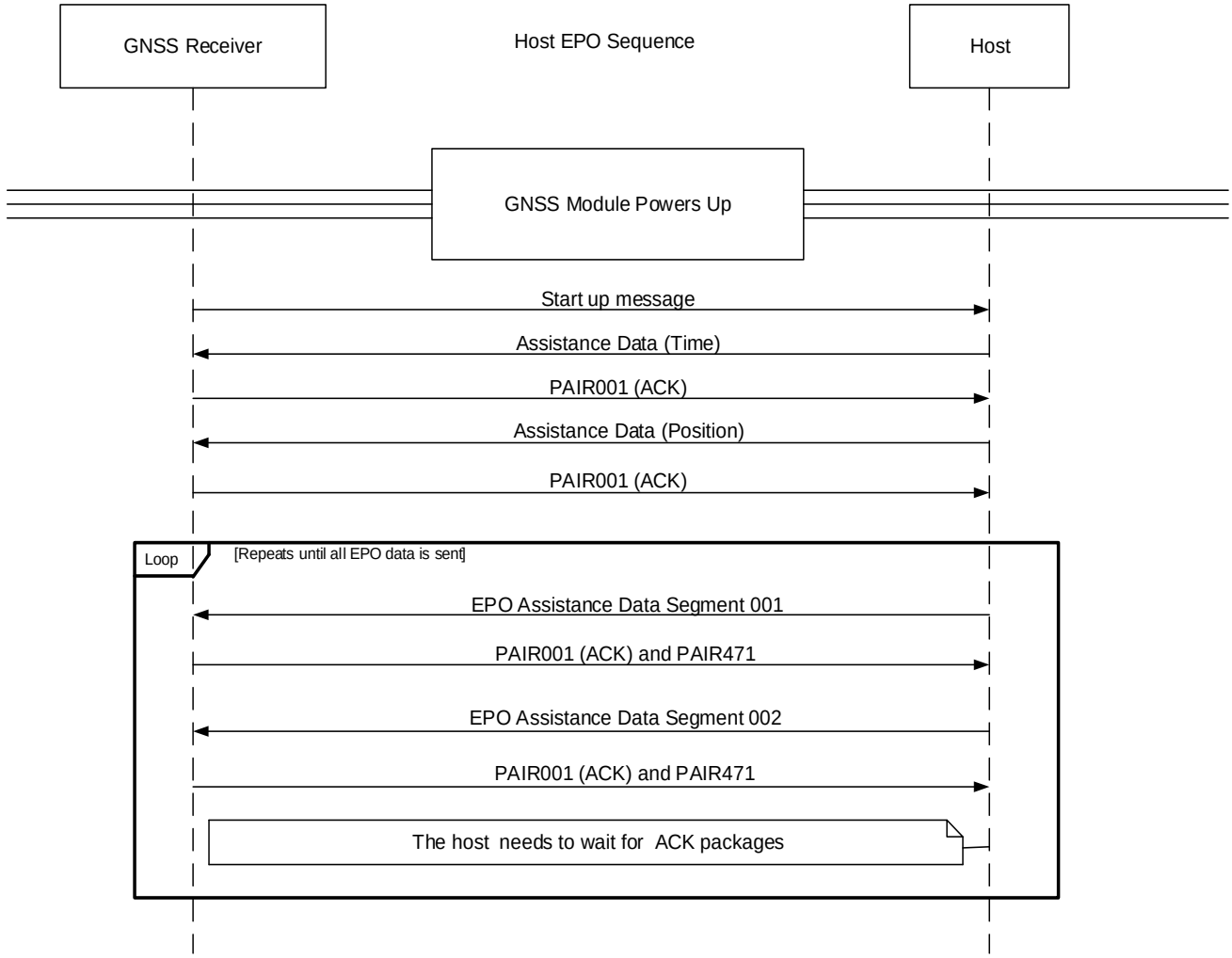

Figure 10: Suggested Sequence for Host EPO

3.2.2. Sample Code to Send EPO

The following is the reference code to send one segment EPO data to GPS chip. It reveals how to construct PAIR messages for GNSS receiver. PAIR messages for Reference Time and Reference Position are not included in this example.

```
#define GNSS_GLONASS_EPO_BASE_ID (64)
#define GNSS_GALILEO_EPO_BASE_ID (100)
#define GNSS_BDS_EPO_BASE_ID (200)

int32_t epo_demo_get_sv_prn(int32_t type, uint8_t *data)
{
    int32_t sv_id, sv_prn = 0;
```

```
    sv_id = data[3];

    switch(type) {
        case EPO_DEMO_MODE_GPS:
            sv_prn = sv_id;
            break;
        case EPO_DEMO_MODE_GLONASS:
            sv_prn = sv_id - GNSS_GLONASS_EPO_BASE_ID;
            break;
        case EPO_DEMO_MODE_GALILEO:
            if(sv_id == 255) {
                sv_prn = 255;
            } else {
                sv_prn = sv_id - GNSS_GALILEO_EPO_BASE_ID;
            }
            break;
        case EPO_DEMO_MODE_BDS:
            if(sv_id == 255) {
                sv_prn = 255;
            } else {
                sv_prn = sv_id - GNSS_BDS_EPO_BASE_ID;
            }
            break;
        default:
            sv_prn = 0;
    }
    return sv_prn;
}

void epo_demo_send_data(epo_demo_epo_data_t *data_p, int32_t data_num, int32_t type){

    char temp_buffer[MNL_SERVICE_MAX_COMMAND_LEN] = {0};
    uint8_t data_buffer[EPO_DEMO_RECORD_SIZE] = {0};
    int32_t i;
    int32_t sv_prn = 0;

    for(i = 0; i < data_num; i++) {
        unsigned int *epobuf = (unsigned int *)data_buffer;
        epo_demo_epo_fread(data_p, data_buffer, EPO_DEMO_RECORD_SIZE);
        sv_prn = epo_demo_get_sv_prn(type, data_buffer);

        sprintf((char *) temp_buffer,
             "471,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X,%X",
               (unsigned int)type,
```

```
                (unsigned int)sv_prn,
                epobuf[0], epobuf[1], epobuf[2], epobuf[3], epobuf[4], epobuf[5],
                epobuf[6], epobuf[7], epobuf[8], epobuf[9], epobuf[10], epobuf[11],
                epobuf[12], epobuf[13], epobuf[14], epobuf[15], epobuf[16], epobuf[17]);

        gnss_app_send_command_ex(temp_buffer);
        memset(temp_buffer, 0, MNL_SERVICE_MAX_COMMAND_LEN);
    }
}
```

4 AGNSS Related Messages

4.1. PAIR001 PAIR_ACK

Acknowledges a PAIR command. An acknowledgement packet **PAIR001** is returned to inform the sender that the receiver has received the packet.

Type:

Output

Synopsis:

```
$PAIR001,<Command_ID>,<Result>*<Checksum>
```

Parameter:

Field	Format	Unit	Description
<Command_ID>	Numeric	-	The type of command/packet to be acknowledged.
<Result>	Numeric	-	0 = The command has been successfully sent 1 = The command is being processed. Please wait for the result 2 = Command sending failed 3 = The command ID is not supported 4 = Command parameter error. Out of range/some parameters were lost/checksum error 5 = The MNL service is busy

Example:

```
$PAIR001,0,3*38
```

4.2. PAIR010 PAIR_REQUEST_AIDING

Notifies the expiration of GNSS aiding data stored in the module.

Type:

Output

Synopsis:

```
$PAIR010,<Type>,<GNSS_System>,<WN>,<TOW>*<Checksum><CR><LF>
```

Parameter:

Field	Format	Unit	Description
<Type>	Numeric	-	Data type. 0 = Need to update EPO data 1 = Need to update the time 2 = Need to update the location
<GNSS_System>	Numeric	-	GNSS data type needed. 0 = Need GPS data 1 = Need GLONASS data 2 = Need Galileo data 3 = Need BDS data 4 = Need QZSS data
<WN>	Numeric	Week	Week Number (accommodating roll-over).
<TOW>	Numeric	Second	Time of Week.

Example:

```
//Please send GPS EPO data when this command is received:
$PAIR010,0,0,2044,369413*33
//Please send reference time when this command is received:
$PAIR010,1,-1*16
//Please send reference location when this command is received:
$PAIR010,2,-1*15
```

NOTE

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

4.3. PAIR470 PAIR_EPO_GET_STATUS

Queries the EPO data status stored in the GPS chip.

Type:

Query

Synopsis:

```
$PAIR470,<System_ID>*<Checksum><CR><LF>
```

Parameter:

Field	Format	Unit	Description
<System_ID>	Numeric	-	0 = GPS 1 = GLONASS 2 = Galileo 3 = BDS

Result:

Returns a **PAIR001** message and the query result.

Query result message format:

```
$PAIR470,<System_ID>,<Set>,<FWN>,<FTOW>,<LWN>,<LTOW>,<FCWN>,<FCTOW>,<LCWN>,<LCTOW>*<Checksum><CR><LF>
```

Parameter included in the result:

Field	Format	Unit	Description
<System_ID>	Numeric	-	0 = GPS 1 = GLONASS 2 = Galileo 3 = BDS
<Set>	Numeric	-	Total number sets of EPO data stored in GNSS chip.
<FWN>	Numeric	-	GPS week number of the first set of EPO data stored in GNSS chip (flash).
<FTOW>	Numeric	-	GPS TOW of the first set of EPO data stored in GNSS chip (flash).
<LWN>	Numeric	-	GPS week number of the last set of EPO data stored in GNSS chip (flash).

<LTOW>	Numeric	-	GPS TOW of the last set of EPO data stored in GNSS chip (flash).
<FCWN>	Numeric	-	GPS week number of the first set of EPO data that are currently used.
<FCTOW>	Numeric	-	GPS TOW of the first set of EPO data that are currently used.
<LCWN>	Numeric	-	GPS week number of the last set of EPO data that are currently used.
<LCTOW>	Numeric	-	GPS TOW of the last set of EPO data that are currently used.

Example:

```
$PAIR470,0*25
$PAIR001,470,0*38
$PAIR470,0,1,2098,194400,2098,216000,2098,194400,2098,216000*38
```

4.4. PAIR471 PAIR_EPO_SET_DATA

Sends the packet containing EPO data for a single satellite.

Type:

Input

Synopsis:

```
$PAIR471,<System_ID>,<SV_ID>,<W[0]>,...,<W[17]>*<Checksum><CR><LF>
```

Parameter:

Field	Format	Unit	Description
<System_ID>	Numeric	-	The GNSS system ID. 0 = GPS 1 = GLONASS 2 = Galileo 3 = BDS
<SV_ID>	Hexadecimal	-	Satellite PRN number for the EPO data to follow. GPS Range: 1–32 GLONASS Range: 1–24 Galileo Range: 1–30 BDS Range: 1–37 Special 255: BDS IONO data.

			Special 254: Galileo IONO data
<W[0]–W[17]>	-	-	18 words [LSB first] of one EPO segment data (total 72 bytes).

Result:

Returns a **PAIR001** message and the query result.

Query result message format:

```
$PAIR471,<System_ID>,<SV_ID>*<Checksum><CR><LF>
```

Example:

```
$PAIR471,1,16,56056272,F2BC0244,4F19AE34,F95C534D,FAE67014,4F19AF6B,F96749BD,9F341F2D,6F4EA9F,77DB4710,66ADAC2,9ADF3B01,8CC8B19C,29D2D20C,FC5B2E94,1000001C,11005000,748B45F4*0A
$PAIR001,471,0*39
$PAIR471,1,16*0E
```

4.5. PAIR472 PAIR_EPO_ERASE_FLASH_DATA

Erases the EPO data stored in the flash memory.

Type:

Command

Synopsis:

```
$PAIR472*<Checksum><CR><LF>
```

Parameter:

None.

Result:

Returns a **PAIR001** message.

Example:

```
$PAIR472*3B
$PAIR001,472,0*3A
```

4.6. PAIR590 PAIR_TIME_SET_REF_UTC

Sends 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.

Type:

Input

Synopsis:

```
$PAIR590,<Year>,<Month>,<Day>,<Hour>,<Minute>,<Second>*<Checksum><CR><LF>
```

Parameter:

Field	Unit	Range	Description
<Year>	Year	> 1980	UTC time: year in 4 digits.
<Month>	Month	1–12	UTC time: month.
<Day>	Day	1–31	UTC time: day.
<Hour>	Hour	0–23	UTC time: hour.
<Minute>	Minute	0–59	UTC time: minute.
<Second>	Second	0–59	UTC time: second.

Result:

Returns a **PAIR001** message.

Example:

```
$PAIR590,2019,2,10,9,0,58*0B
$PAIR001,590,0*37
```

4.7. PAIR600 PAIR_LOC_SET_REF

Sends reference location to GNSS chip for faster TTFF.

Type:

Input

Synopsis:

```
$PAIR600,<Lat>,<Lon>,<Height>,<AccMaj>,<AccMin>,<Bear>,<AccVert>*<Checksum><CR><LF>
```

Parameter:

Field	Unit	Range	Description
<Lat>	Degree	-90.0–90.0	WGS84 geodetic latitude. It's recommended to express this value in floating-point with 6 decimal points.
<Lon>	Degree	-180.0–180.0	WGS84 geodetic longitude. It's recommended to suggest to express this value in floating-point with 6 decimal points.
<Height>	m	-	WGS84 ellipsoidal altitude.
<AccMaj>	m	> 0	Horizontal uncertainty semi-major axis.
<AccMin>	m	> 0	Horizontal uncertainty semi-minor axis.
<Bear>	Degree	0–179	Error ellipse semi-major axis bearing.
<AccVert>	m	> 0	Vertical uncertainty.

Result:

Returns a **PAIR001** message.

Example:

```
$PAIR600,24.772816,121.022636,175.0,50.0,50.0,0.0,100.0*06
$PAIR001,600,0*3D
```

5 EPO Usage Through QGNSS

QGNSS is a Quectel official tool which allows users to evaluate the receiver performance as well as to perform different measurements on the receiver. Visit *http://220.180.239.212:8177/* for details on QGNSS. This chapter describes how to evaluate the EPO functionality through QGNSS.

5.1. Download Flash EPO with QGNSS

Steps to download Flash EPO with the QGNSS tool:

1. Run the QGNSS tool.

2. In the main interface, click "**AGNSS**" → "**Assistant GNSS Offline"** as shown below.

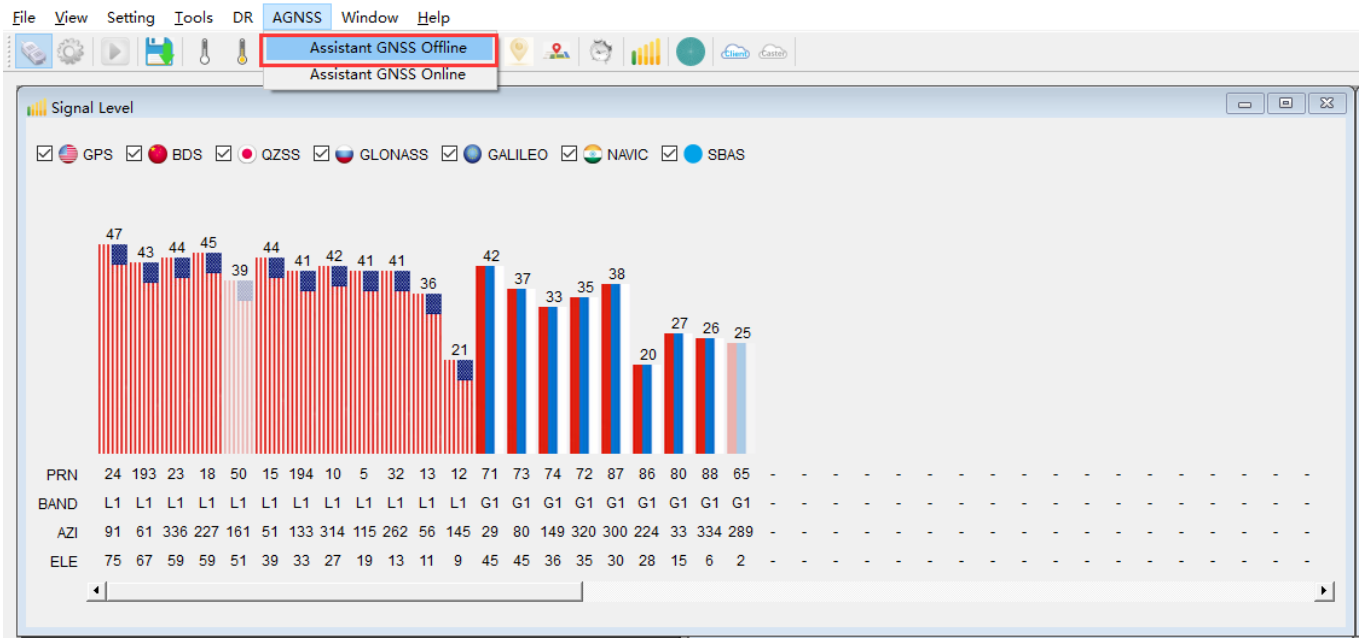

Figure 11: Flash EPO Setting Interface of QGNSS

3. Download EPO file to the module.
   a) Click the "**Connect**" button to connect to the FTP server.
   b) Select EPO file.
   c) Click the "**Download selected file**" button to download the EPO file to computer.
   d) Select Satellites type.

e) Click the "…" button to select EPO file.
f) Click the "**Download**" to download the EPO file to module.

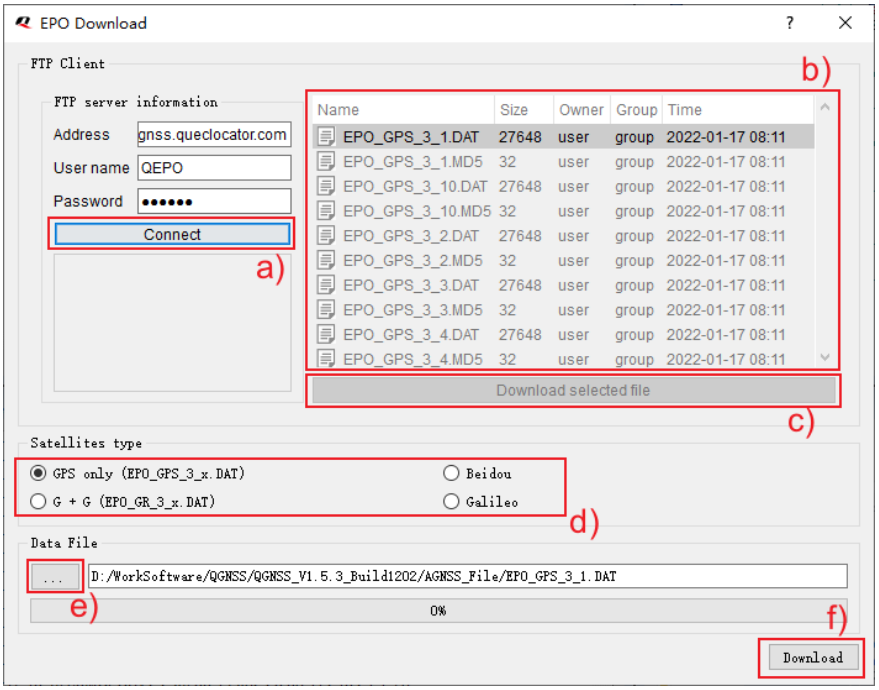

Figure 12: EPO File Downloading

5.2. Download Host EPO with QGNSS

The steps to download Host EPO with the QGNSS tool:

1. Run QGNSS tool.

2. In the main interface, click "**AGNSS**" → "**Assistant GNSS Online"** as shown below.

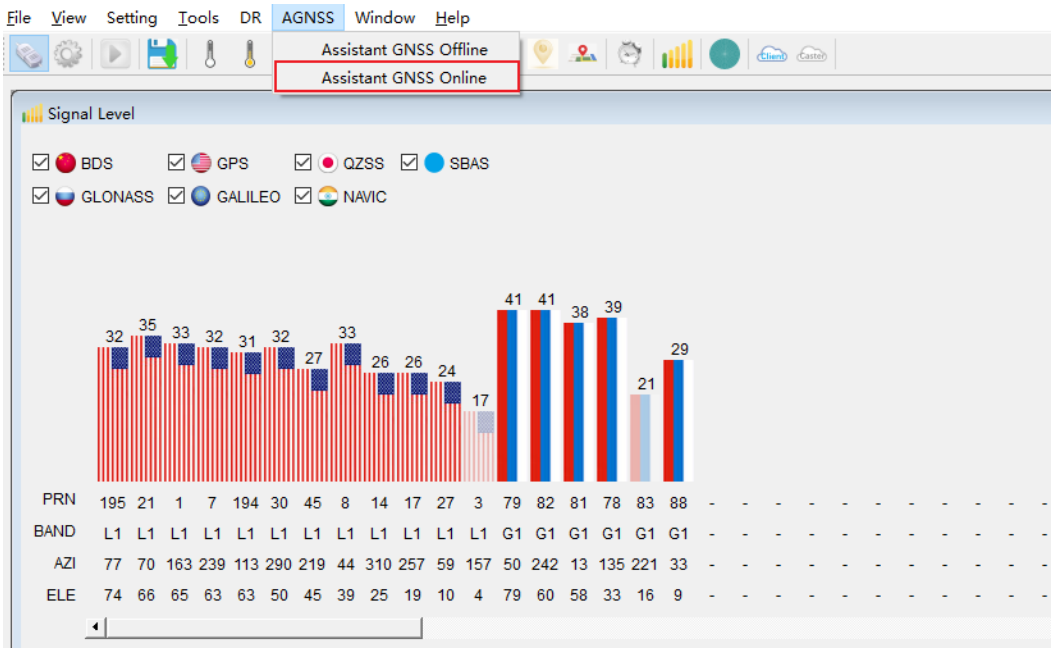

Figure 13: Host EPO Setting Interface of QGNSS

3. Configure parameter:
   a) Check "**Use Current Position**" to use current position**.**
   b) Check "**Use Current UTC**" to use current time**.**
   c) Click "**Transfer**" to download host EPO file.

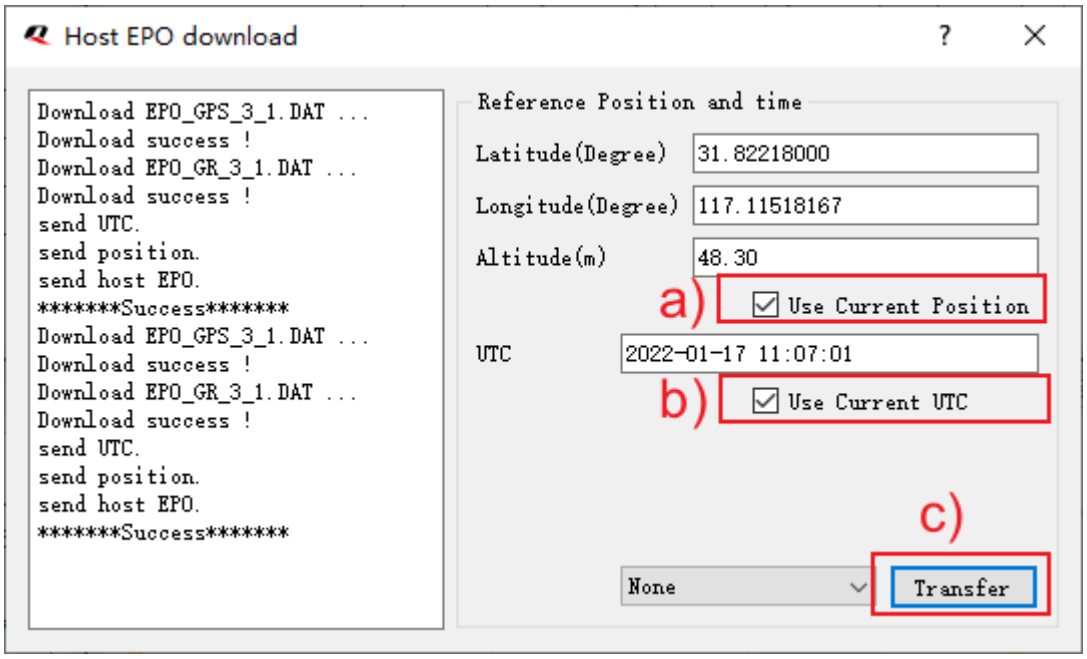

Figure 14: Host EPO File Downloading

6 AGNSS Implementation Example

This chapter gives examples to show how EPO files can be downloaded to the module.

6.1. Flash EPO Implementation

Blue: Send data
Red: ACK information

//Host sends **$PAIR472*3B** to erase the EPO data stored in the flash memory:
$PAIR472*3B

//Module returns a **PAIR001** message:
$PAIR001,472,0*3A

//Host sends EPO start message in hex format:
04 24 B0 04 01 00 52 E7 AA 44

//Module returns an ack message:
04 24 E8 03 04 00 B0 04 00 00 5B AA 44

//Host sends EPO data in hex format:
04 24 B1 04 48 00 BF 96 05 41 6E 3A 74 05 C3 21…….AA 44

//Module returns an ack message:
04 24 E8 03 04 00 B1 04 00 00 5A AA 44

//Host sends EPO data in hex format:
04 24 B1 04 48 00 BF 96 05 42 09 3A 74 0C CA 77……. AA 44

//Module returns an ack message:
04 24 E8 03 04 00 B1 04 00 00 5A AA 44
……

//Host sends EPO end data in hex format:
04 24 B2 04 01 00 52 E5 AA 44

//Module returns an ack message:
04 24 E8 03 04 00 B2 04 00 00 59 AA 44

//Host queries the EPO data status stored in the GPS chip:

$PAIR470,0*25

//Module returns **PAIR001** and **PAIR470** messages:
$PAIR001,470,0*38
$PAIR470,0,1,2098,194400,2098,216000,2098,194400,2098,216000*38

6.2. Host EPO Implementation

Blue: Send data
Red: ACK information

//Host sends the power-on GNSS system command **PAIR002**:
$PAIR002*38

//Module returns **PAIR001**:
$PAIR001,002,0*39

//Module outputs **PAIR010** messages automatically:
$PAIR010,1,-1*16
$PAIR010,2,-1*15

//Host sends the current UTC time command **PAIR590**:
$PAIR590,2021,10,18,08,59,00*3B

//Module returns a **PAIR001** message:
$PAIR001,590,0*37

//Host sends the current location information command **PAIR600**:
$PAIR600,31.822203,117.115219,175.0,50.0,50.0,0.0,100.0*0F

//Module returns a **PAIR001** message:
$PAIR001,600,0*3D

//Host sends EPO data:
$PAIR471,0,1,10596C0,A174051A,1B2EDE67,9F0BB6,17C37A4,1B2EDE22,F85B368E,845FB0C9,6F18C40,23557111,2A4CBD5,A60348AB,FEF7E24,2F236B88,2439FDC6,1000001C,0,4860BF93*44

//Module returns **PAIR001** and **PAIR471** messages:
$PAIR001,471,0*39
$PAIR471,10596C0*5C

//Host sends EPO data:

$PAIR471,0,2,20596C0,F0740341,1B2EEE36,5F3FAA,17E07E6,1B2E1100,F8C1048F,84A50985,6F1BD33,29192005,D651ED6,A600506D,256C95F,20430E67,C36C910D,1000001C,0,FAC0DA552A 33 430D 0A*3C

//Module returns **PAIR001** and **PAIR471** messages:
$PAIR001,471,0*39
$PAIR471,20596C0*5F
……

//Host sends EPO data:
$PAIR471,0,7,70596C0,377403C2,1B2E41F1,F757006,C57EB,1B2EBF14,F89F543F,8986E880,6F1E693,DC3908E,DA7BB7,A603A757,8FBA37BF,21C8A8F0,A2247EAD,1000001C,22000000,DDACBBB6*0B

//Module returns **PAIR001** and **PAIR471** messages:
$PAIR001,471,0*39
$PAIR471,C30596BE*69

7 Appendix References

Table 10: Terms and Abbreviations

Abbreviation	Description
ACK	Acknowledgement
AGNSS	Assisted GNSS (Global Navigation Satellite System)
EPO	Extended Prediction Orbit
GLONASS	Global Navigation Satellite System
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMEI	International Mobile Equipment Identity
MNL	MTK Navigation Lib
NACK	Negative Acknowledgement
RAM	Random Access Memory
SV	Satellite Vehicle
TOW	Time of week
TTFF	Time to First Fix
UART	Universal Asynchronous Receiver/Transmitter
UTC	Coordinated Universal Time
URL	Uniform Resource Locator