LC26G (AB)&LC76G&LC86G Series

Firmware Upgrade Guide

GNSS Module Series

Version: 1.0

Date: 2022-11-04

Status: Released

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	**LC26G (AB)&LC76G&LC86G Series Firmware Upgrade Guide**
Subtitle	GNSS Module Series
Document Type	Firmware Upgrade Guide
Document Status	Released

Revision History

Version	Date	Description
-	2022-08-11	Creation of the document
1.0	2022-11-04	First official release

Contents

Table Index

Figure Index

1 Introduction

This document introduces the procedure for upgrading the firmware of the Quectel LC26G (AB), LC76G and LC86G series GNSS modules.

Following the procedure illustrated in this document, you can upgrade the firmware to the target GNSS module via the UART interface. The baud rate for firmware upgrade is 115200 bps by default.

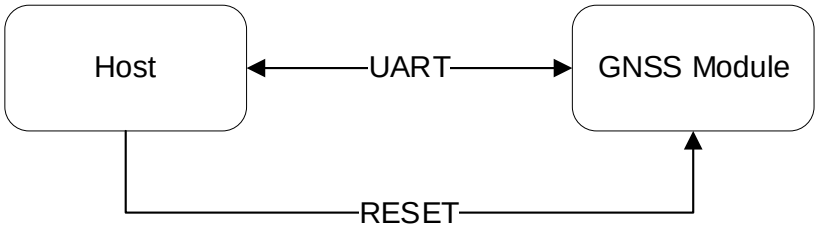

Figure 1: Firmware Upgrade Connection

This document is applicable to the following Quectel GNSS modules:

Table 1: Applicable Modules

Module	Variant
LC26G (AB)	LC26G (AB)
LC76G	LC76G (AB)
	LC76G (PA)
	LC76G (PB)
LC86G	LC86G (AA)
	LC86G (AB)
	LC86G (LA)

NOTE

1. If the firmware upgrade procedure fails or it is stopped by the user, no backup firmware can be executed; the only way to reprogram a non-working firmware is to reset the hardware and start a new upgrade procedure.
2. Please be aware that firmware upgrade can be done as well by using a PC's software utility. For more details, see *document [1], [2] and [3] EVB user guide*.

2 Firmware Upgrade Procedure

This chapter describes all the necessary steps in the firmware upgrade procedure.

The following table defines all the constants used in this document.

Table 2: Constant List

Constant Name	Constant Value	Steps
BROM_ERROR	0x1000	*Figure 4: Disable Module WDT* *Figure 5: Modify Baud Rate* *Figure 6: Send DA File to Module* *Figure 7: Jump to DA*
Module WDT Register Address 1	0xA2080000	*Figure 4: Disable Module WDT*
WDT Value 1	0x0010	
Module WDT Register Address 2	0xA2080030	
WDT Value 2	0x0040	
DA Run/Start Address	0x04200000	*Figure 6: Send DA File to Module* *Figure 7: Jump to DA*
Flash Start Address	0x08000000	*Figure 9: Sync with DA and Get DA Report*
Flash Size	0x00200000	
Format Physical Address	0x08000000	*Figure 10: Format Flash*
Format Length	0x1F8000	
FW Flash Address	0x08000000 (partition_table.bin) 0x08001000 (bootloader.bin) 0x08009000 (*<Version>*.bin) 0x081F7000 (gnss_config.bin)	*Figure 11: Send FW File to Module*
FW Packet Length	0x00001000	

2.1. Communication Between Host and Module

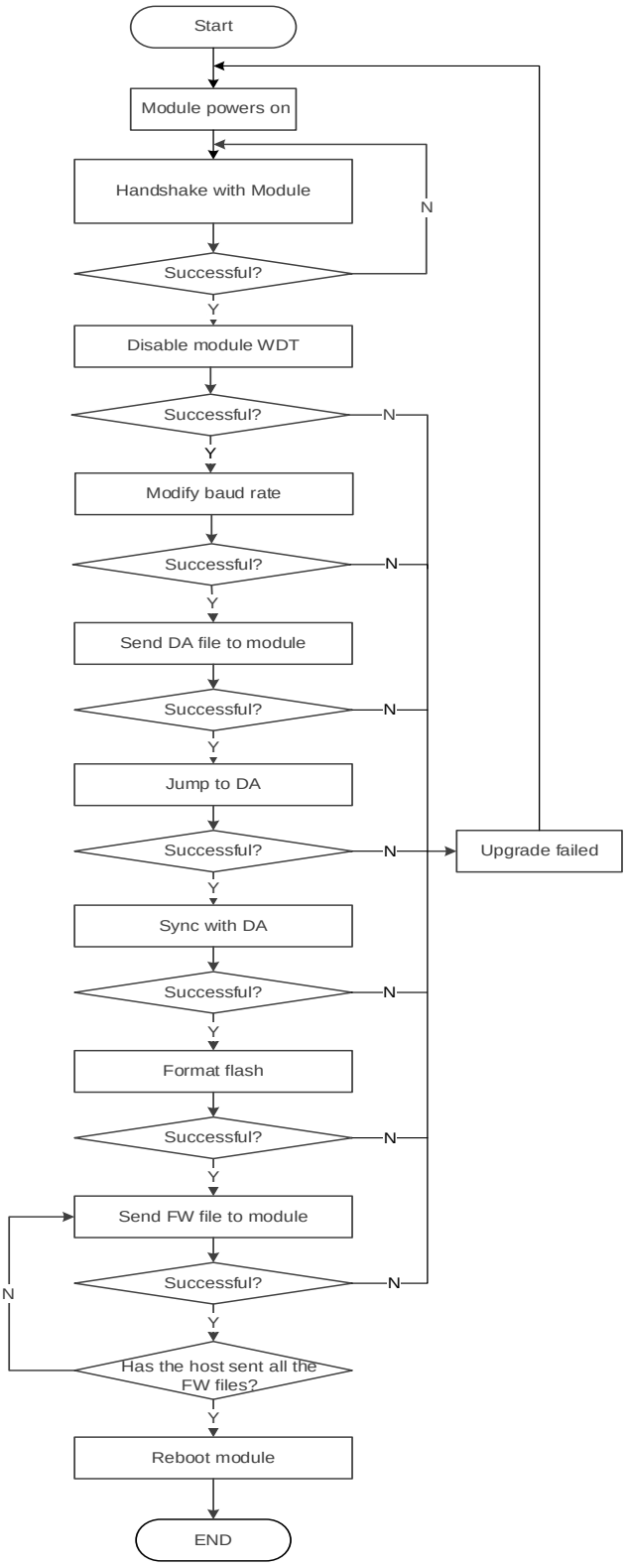

Figure 2: Communication Between Host and Module

The following sections illustrate how the host establishes communication with the module and sends commands and files to the module:

- Handshake with the module
- Disable module WDT (Watch Dog Timer)
- Modify baud rate
- Send DA (Download Agent) file to the module
- Jump to DA
- Synchronize with DA
- Format flash
- Send FW (firmware) file to the module

2.2. Host and BROM Procedure

2.2.1. Handshake with Module

This section describes the handshake mechanism for the communication between the host and the module.

Reset the module after it is powered on and receives **0xA0** from the host. If the module does not receive **0xA0** in 150 milliseconds after being reset, the handshake process will fail.

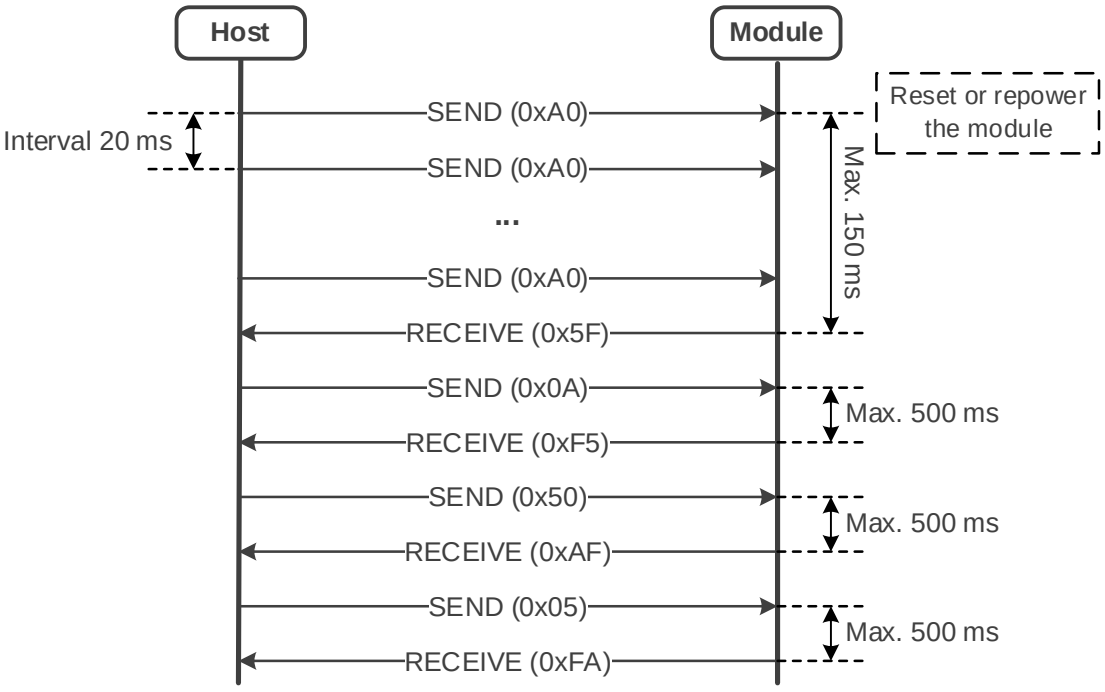

Figure 3: Handshaking

NOTE

1. The host sends **0xA0** repeatedly until it receives **0x5F** from the module, after which it sends **0x0A** to the module.
2. The baud rate should be 115200 bps during handshaking.

2.2.2. Disable Module WDT

The following diagram illustrates how to disable module WDT to avoid module reboot. For the constant parameters in the figure below, see *Table 2: Constant List*.

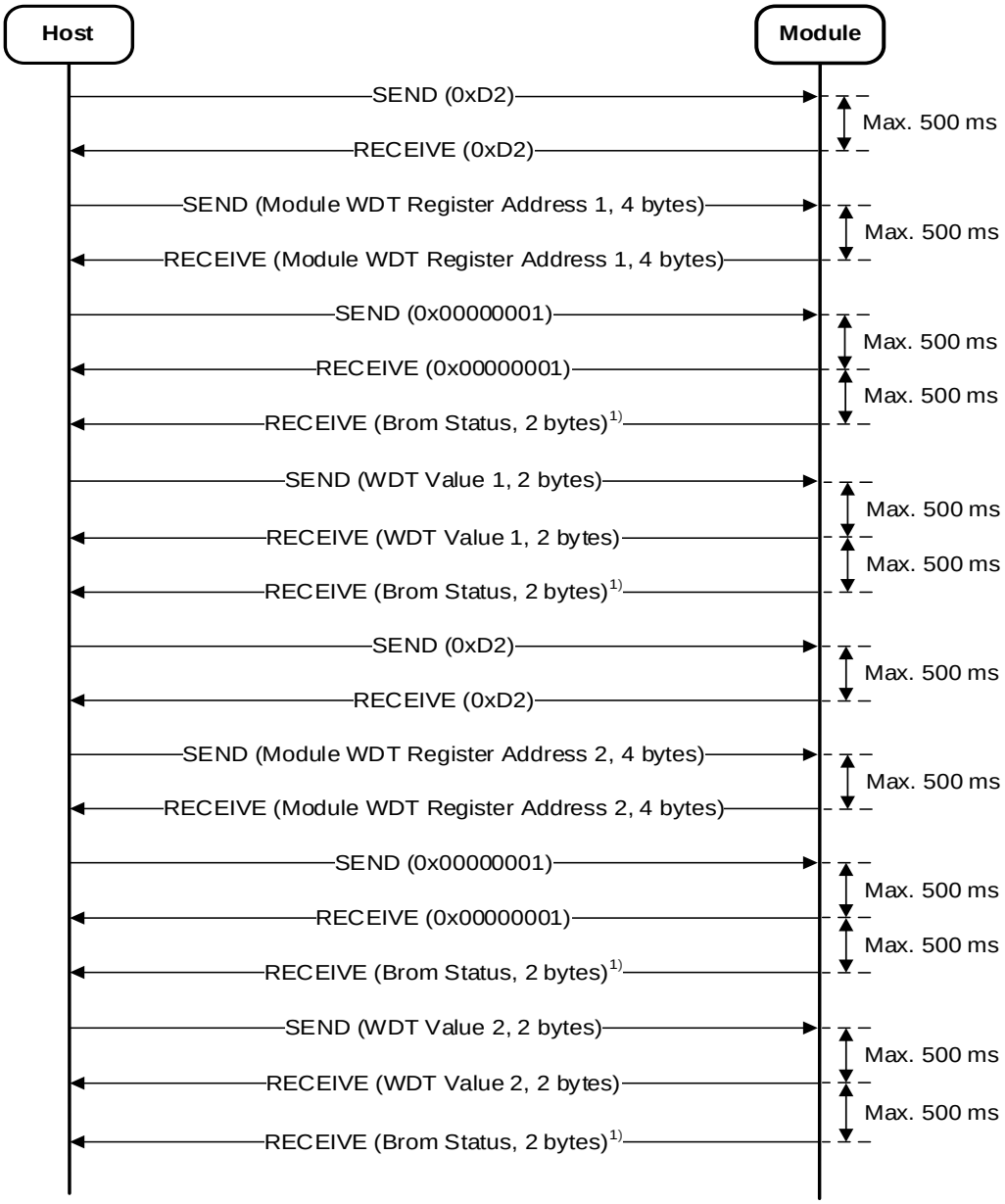

Figure 4: Disable Module WDT

NOTE

[1] The value of Brom status must be less than BROM_ERROR, otherwise this process will fail.

2.2.3. Modify Baud Rate

This section describes how the host notifies the module to modify the baud rate. After the host receives Brom Status from the module, it is necessary to manually modify host baud rate. Different baud rates correspond to different DA files, and you need to set the baud rate according to the corresponding DA file.

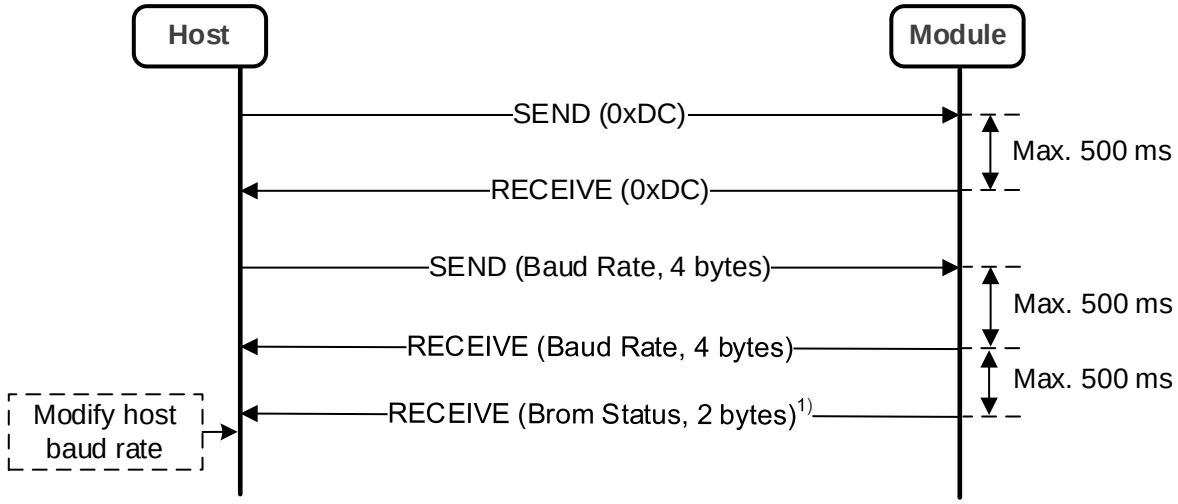

Figure 5: Modify Baud Rate

NOTE

1. [1] The value of Brom status must be less than BROM_ERROR, otherwise this process will fail.
2. If the DA file with baud rate of 115200 is used for upgrading, this step can be omitted.

2.2.4. Send DA File to Module

The following diagram illustrates how to send a DA file to the module. For the constant parameters in the figure below, see *Table 2: Constant List*.

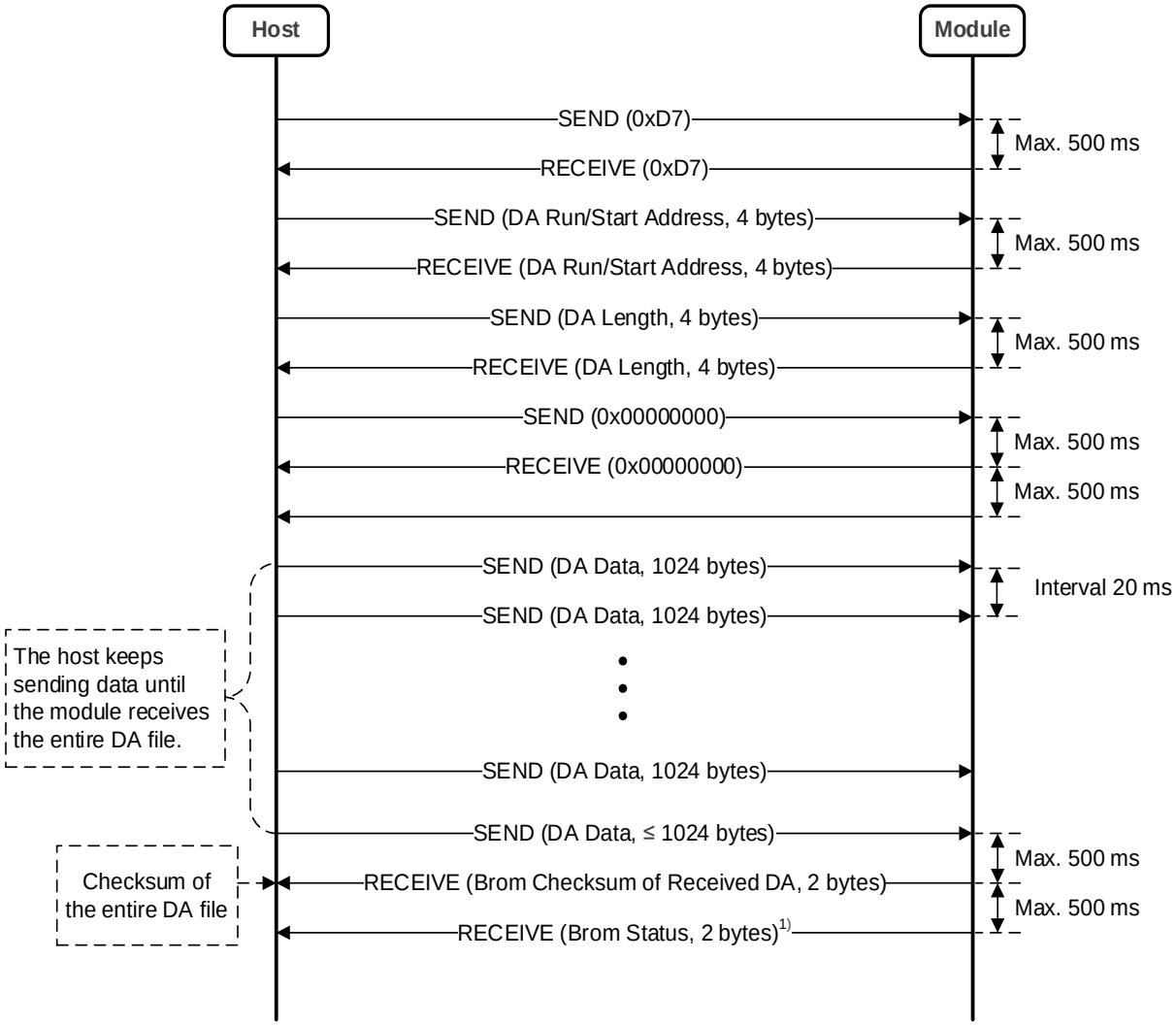

Figure 6: Send DA File to Module

NOTE

1) The value of Brom status must be less than BROM_ERROR, otherwise this process will fail.

Sample Code of DA File Checksum

```
uint16_t DA_compute_checksum (uint8_t *buf, uint32_t buf_len)
{
    uint16_t checksum = 0;
    if (buf == NULL || buf_len == 0) {
        return 0;
```

```
    }
    int i = 0;
    for (i = 0; i < buf_len / 2; i++) {
        checksum ^= *(uint16_t *) (buf + i * 2);
    }
    if ((buf_len % 2) == 1) {
        checksum ^= buf[i * 2];
    }
    return checksum;
}
uint16_t local_check_sum ^= DA_compute_checksum(data_buf, len);
```

2.2.5. Jump to DA

The following diagram illustrates how to inject the DA file and how to execute it. For the constant parameters in the figure, see *Table 2: Constant List*.

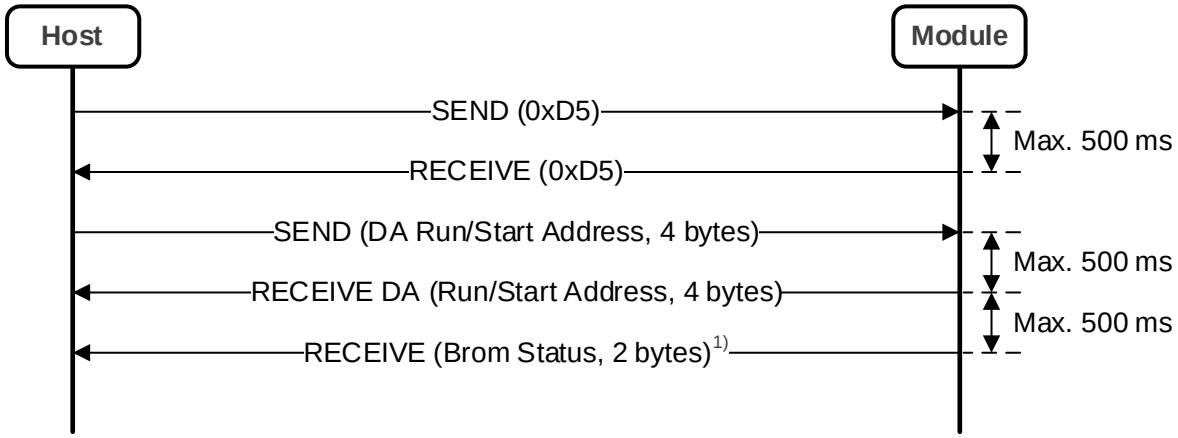

Figure 7: Jump to DA

NOTE

1) The value of Brom status must be less than BROM_ERROR, otherwise this process will fail.

2.3. Host and DA Procedure

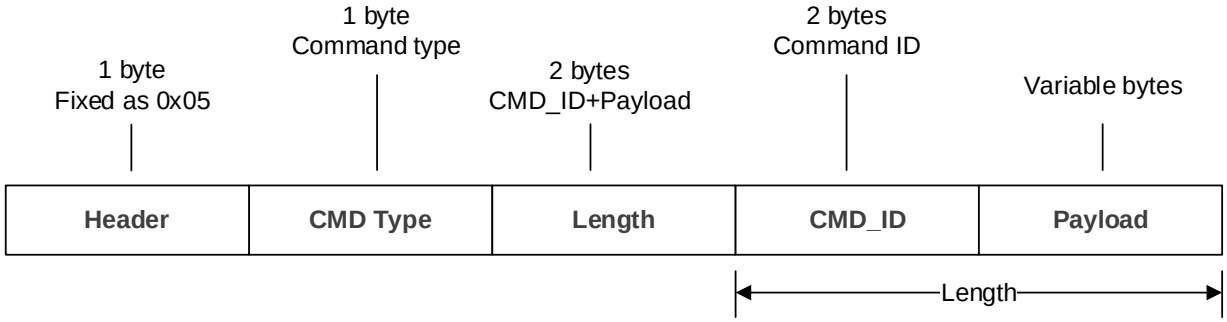

Figure 8: Binary Protocol Structure

Table 3: Description of Binary Protocol Fields

Field	Length (Byte)	Description
Header	1	Fixed as 0x05.
CMD Type	1	Command type. Sent type: 0x5A. Response type: 0x5B.
Length	2	Length of CMD_ID and Payload.
CMD_ID	2	Command ID.
Payload	Variable	Payload data to be transferred.

Sample Code for FW Packet and Format Address Checksum

```
//Generate CRC Table

uint32_t crc_table[256];

void Make_CRC32table(void)
{
    uint32_t value;
    uint16_t i,j;
    for(i=0;i<256;i++)
    {
        value=i;
        for(j=0;j<8;j++)
```

```
        {
            if(value&1)
            {
                value=(value>>1)^0xedb88320;
            }
            else
            {
                value=(value>>1);
            }
        }
        crc_table[i]=value;
    }
}

//Calculate CRC on a buffer of known length

uint32_t Calculate_CRC32(uint8_t*buffer,uint16_tlength)
{
    uint32_t crc32=0xFFFFFFFF;
    while(length--)
    {
        crc32=crc_table[(crc32&0xFF)^*buffer++]^(crc32>>8);
    }
    return crc32^0xFFFFFFFF;
}
```

NOTE

1. Messages are in little endian format.
2. Use CRC32 checksum (0xEDB88320).

2.3.1. Synchronize with DA

The following diagram illustrates how to synchronize with DA and get the DA report. Before writing to flash, we need to get flash information, including the Flash Start Address, Flash Size, and Flash ID.

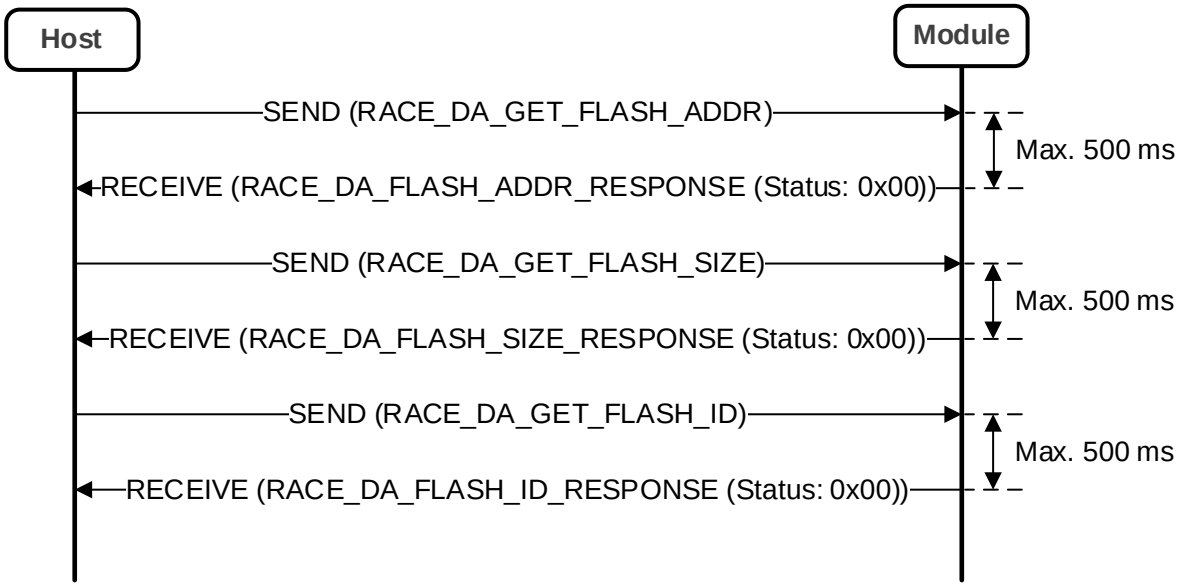

Figure 9: Sync with DA and Get DA Report

2.3.1.1. Get Flash Address

RACE_DA_GET_FLASH_ADDR

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5A	0x0002	0x210E	-

RACE_DA_FLASH_ADDR_RESPONSE

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5B	0x0007	0x210E	See *Table 4: RACE_DA_FLASH_ADDR_RESPONSE Payload*.

Table 4: RACE_DA_FLASH_ADDR_RESPONSE Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	1	Status	-	Response status. 0x00 = Success. Others = Fail.
1	4	Flash Start Address	-	Flash start address.

2.3.1.2. Get Flash Size

RACE_DA_GET_FLASH_SIZE

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5A	0x0002	0x210F	-

RACE_DA_FLASH_SIZE_RESPONSE

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5B	0x0007	0x210F	See *Table 5: RACE_DA_FLASH_SIZE_RESPONSE Payload*.

Table 5: RACE_DA_FLASH_SIZE_RESPONSE Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	1	Status	-	Response status. 0x00 = Success. Others = Fail.
1	4	Flash Size	Byte	Flash size.

2.3.1.3. Get Flash ID

RACE_DA_GET_FLASH_ID

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5A	0x0002	0x2110	-

RACE_DA_FLASH_ID_RESPONSE

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5B	0x0006	0x2110	See *Table 6: RACE_DA_FLASH_ID_RESPONSE Payload*.

Table 6: RACE_DA_FLASH_ID_RESPONSE Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	1	Status	-	Response status. 0x00 = Success.

Byte Offset	Length (Byte)	Name	Unit	Description
				Others = Fail.
1	3	Flash ID	-	FlashID[3] (1*3 bytes).

2.3.2. Format Flash

The following diagram illustrates how to format the module's flash. Format Start Address should be 4 KB aligned, and format Length should be 4 KB/64 KB.

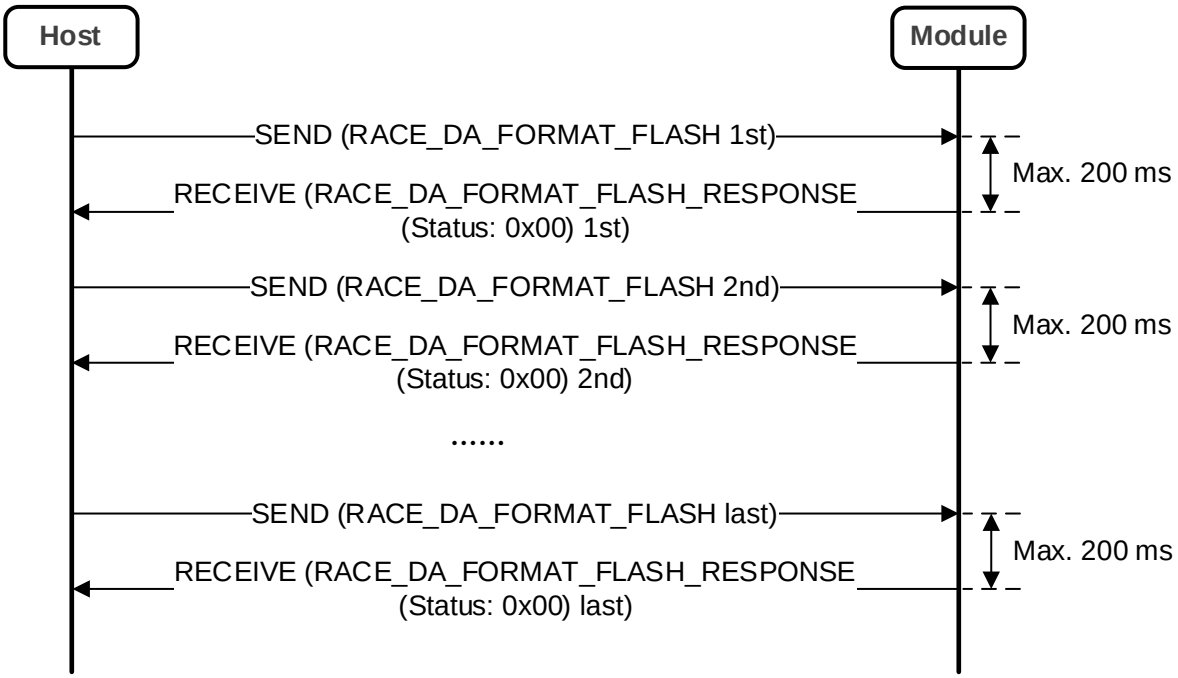

Figure 10: Format Flash

2.3.2.1. Format Flash Address

RACE_DA_FORMAT_FLASH

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5A	0x000E	0x2104	See *Table 7: RACE_DA_FORMAT_FLASH Payload*.

RACE_DA_FORMAT_FLASH_RESPONSE

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5B	0x0007	0x2104	See *Table 8: RACE_DA_FORMAT_FLASH_RESPONSE Payload*.

Table 7: RACE_DA_FORMAT_FLASH Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	4	Format Start Address	-	Flash start address of each erase.
4	4	Format Length	Byte	Length of each erase.
8	4	CRC32	-	CRC32 checksum.

Table 8: RACE_DA_FORMAT_FLASH_RESPONSE Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	1	Status	-	Response status. 0x00 = Success. Others = Fail.
1	4	Format Start Address	-	Start address of each erase.

NOTE

Format address range: 0x08000000–0x081F8000.

2.3.3. Send FW File to Module

The following diagram illustrates how to send FW file to module. Data Start Address should be 4 KB alignment.

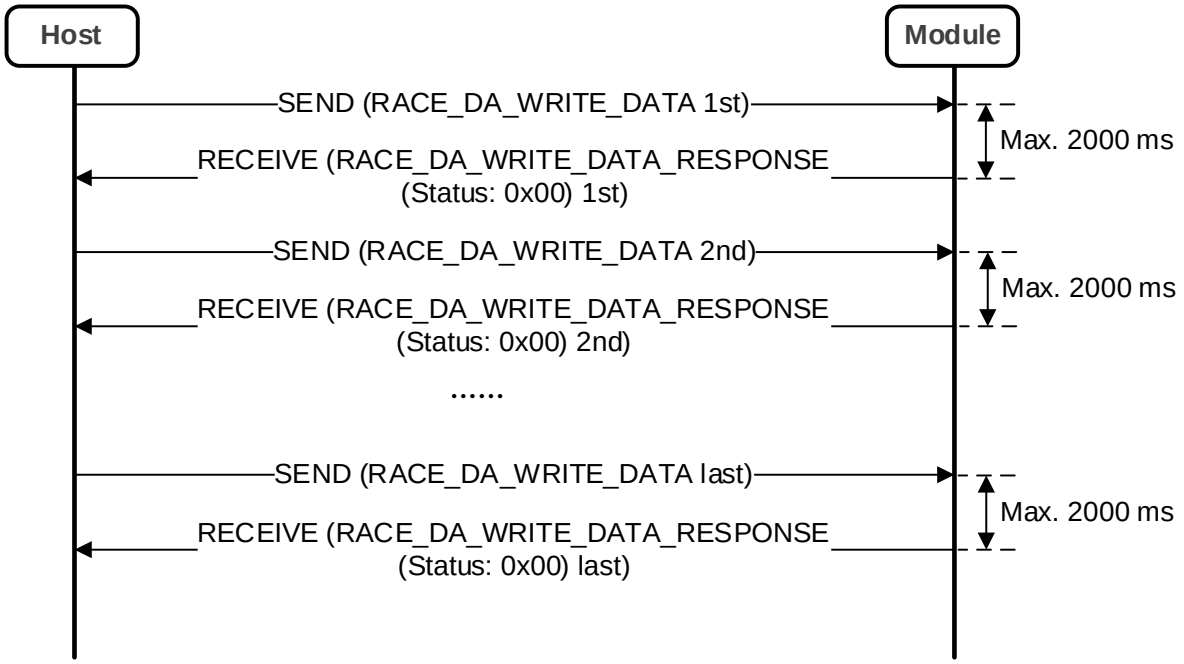

Figure 11: Send FW File to Module

2.3.3.1. Send FW Packet

RACE_DA_WRITE_DATA

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5A	0x100C	0x2100	See *Table 9: RACE_DA_WRITE_DATA Payload*.

RACE_DA_WRITE_DATA_RESPONSE

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5B	0x0007	0x2100	See *Table 10: RACE_DA_WRITE_DATA_RESPONSE Payload*.

Table 9: RACE_DA_WRITE_DATA Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	4	Data Start Address	-	Start address of each write.
4	2	Data Length	Byte	FW packet Length.

Byte Offset	Length (Byte)	Name	Unit	Description
6	4096	Data	-	Data of FW packet.
4102	4	CRC32	-	CRC32 checksum.

Table 10: RACE_DA_WRITE_DATA_RESPONSE Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	1	Status	-	Response status. 0x00 = Success. Others = Fail.
1	4	Data Start Address	-	Start address of each write.

NOTE

If the last FW packet is less than 4096 bytes, add "**0xFF**" at the end of the last FW packet.

2.3.4. Auto Reboot

After the module firmware is upgraded successfully, the host sends this protocol to reboot the module. The following diagram illustrates how to reboot automatically.

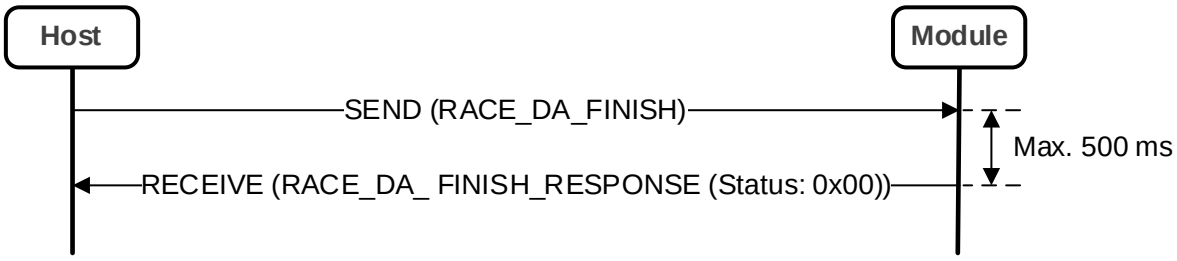

Figure 12: Flow of Auto Reboot

2.3.4.1. Reboot Module

RACE_DA_FINISH

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5A	0x0003	0x2106	See *Table 11: RACE_DA_FINISH Payload*.

RACE_DA_FINISH_RESPONSE

Header	CMD Type	Length	CMD_ID	Payload
0x05	0x5B	0x0003	0x2106	See *Table 12: RACE_DA_FINISH_RESPONSE Payload*.

Table 11: RACE_DA_FINISH Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	1	Reboot Flag	-	Fixed as 0x01.

Table 12: RACE_DA_FINISH_RESPONSE Payload

Byte Offset	Length (Byte)	Name	Unit	Description
0	1	Status	-	Response status. 0x00 = Success. Others = Fail.

NOTE

The module will output firmware version information automatically after each boot-up. For example, LC76G (AB) will output **$PQTMVER,MODULE_LC76GABNR02A01S,2022/09/14,11:47:03*79** after its boot-up. You can also execute **$PQTMVERNO*58** to query the firmware version information. For more details, see *document [4] protocol specification*.

3 Upgrade Implementation Example

Below is an example of the LC76G (AB) firmware upgrade.

//*Figure 3: Handshaking*
//Host keeps sending **0xA0**. Reset the module.
A0
A0
A0
…
A0
A0
//Module responds with **0x5F**.
5F
//Host sends **0x0A**.
0A
//Module responds with **0xF5**.
F5
//Host sends **0x50**.
50
//Module responds with **0xAF**.
AF
//Host sends **0x05**.
05
//Module responds with **0xFA**.
FA
//*Figure 4: Disable Module WDT*
//Host sends **0XD2**.
D2
//Module responds with **0xD2**.
D2
//Host sends **0xA2080000**.
A2 08 00 00
//Module responds with **0xA2080000**.
A2 08 00 00
//Host sends **0x00000001**.
00 00 00 01
//Module responds with **0x00000001**.
00 00 00 01

//Module responds with Brom status.
00 01
//Host sends **0x0010**.
00 10
//Module responds with **0x0010**.
00 10
//Module responds with Brom status.
00 01
//Host sends **0XD2**.
D2
//Module responds with **0xD2**.
D2
//Host sends **0xA2080030**.
A2 08 00 30
//Module responds with **0xA2080030**.
A2 08 00 30
//Host sends **0x00000001**.
00 00 00 01
//Module responds with **0x00000001**.
00 00 00 01
//Module responds with Brom status.
00 01
//Host sends **0x0040**.
00 40
//Module responds with **0x0040**.
00 40
//Module responds with Brom status.
00 01
//
Figure 5: Modify Baud **Rate** (If the DA file with baud rate of 115200 is used to upgrade, this step can be omitted.)
//Host sends **0xDC**.
DC
//Module responds with **0xDC**.
DC
//Host sends **0x000E1000**.
00 0E 10 00
//Module responds with **0x000E1000**.
00 0E 10 00
//Module responds with Brom status.
00 00
//
Figure 6: Send DA File to Module
//Host sends **0xD7**.

D7
//Module responds with **0xD7**.
D7
//Host sends DA run/start address.
04 20 00 00
//Module responds with DA run/start address.
04 20 00 00
//Host sends DA length.
00 00 AB B4
//Module responds with DA length.
00 00 AB B4
//Host sends **0x00000000**.
00 00 00 00
//Module responds with **0x00000000**.
00 00 00 00
//Module responds with Brom status.
00 00
//Host sends DA data, in a chunk of 1024 bytes.
DF F8 58 D0 72 B6 16 48 80 47 16 49 17 4B 17 4C 18 4A 42 F8 04 0B 42 F8 ……
//Host sends DA data, in a chunk of 1024 bytes.
0B 28 00 F2 30 81 DF E8 10 F0 0C 00 2F 00 53 00 76 00 98 00 BB 00 DE 00 ……
……
//Host sends the last chunk of DA data. (length ≤ 1024 bytes)
00 00 00 00 00 00 00 00 00 00 40 00 01 00 00 00 00 00 00 00 00 00 00 00 ……
//Module responds with Brom checksum of the received DA (2 bytes). (*Sample Code of DA File Checksum*)
66 11(DA file checksum with baud rate of 921600. If the baud rate is 115200, DA file check value is 65 1F.)
//Module responds with Brom status.
00 00
//*Figure 7: Jump to DA*
//Host sends **0xD5**.
D5
//Module responds with **0xD5**.
D5
//Host sends DA run/start address.
04 20 00 00
//Module responds with DA run/start address.
04 20 00 00
//Module responds with Brom status.
00 00
//*Figure 9: Sync with DA and Get DA Report*
//Host requests the Flash Start Address.
05 5A 02 00 0E 21

//Module responds to the Flash Start Address in the DA file.
05 5B 07 00 0E 21 00 00 00 00 08
//Host requests the Flash Size.
05 5A 02 00 0F 21
//Module responds to Flash Size.
05 5B 07 00 0F 21 00 00 00 20 00
//Host requests the Flash ID.
05 5A 02 00 10 21
//Module responds with Flash ID.
05 5B 06 00 10 21 00 C8 60 15
//*Figure 10: Format Flash*
//Host sends format command to the module. (Format Length: 64 KB) (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0E 00 04 21 00 00 00 08 00 00 01 00 F9 D9 F6 3A
//Module acknowledges that it executed the format command successfully.
05 5B 07 00 04 21 00 00 00 00 08
//Host sends format command to the module. (Format length: 64 KB) (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0E 00 04 21 00 00 01 08 00 00 01 00 5C 0A AA F1
//Module acknowledges that it executed the format command successfully.
05 5B 07 00 04 21 00 00 00 01 08
……
//Host sends format command to the module. (Remaining Length ≤ 64 KB, Format Length: 4 KB) (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0E 00 04 21 00 00 1F 08 00 10 00 00 86 FB 4B CD
//Module acknowledges that it executed the format command successfully.
05 5B 07 00 04 21 00 00 00 1F 08
//Host sends format command to the module. (Remaining Length ≤ 64 KB, Format Length: 4 KB) (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0E 00 04 21 00 00 1F 08 00 10 00 00 48 67 95 AA
//Module acknowledges that it executed the format command successfully.
05 5B 07 00 04 21 00 00 10 1F 08
……
//Host sends format command to the module. (Remaining Length ≤ 64 KB, Format Length: 4 KB) (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0E 00 04 21 00 70 1F 08 00 10 00 00 AD 2B 23 21
//Module acknowledges that it executed the format command successfully.
05 5B 07 00 04 21 00 00 70 1F 08
//*Figure 11: Send FW File to Module*
//Host sends FW packet (*patition_table.bin* 4096 bytes). (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0C 10 00 21 00 00 00 08 00 10 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF 00 ……
//Module responds with the received FW packet.
05 5B 07 00 00 21 00 00 00 00 08

//Host sends FW packet (*bootloader.bin* 4096 bytes). (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0C 10 00 21 00 10 00 08 00 10 DF F8 3C D0 72 B6 0F 48 0F 49 01 60 00 20 03 21 0E ……
//Module responds with the received FW packet.
05 5B 07 00 00 21 00 00 10 00 08
//Host sends FW packet (*bootloader.bin* 4096 bytes). (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0C 10 00 21 00 20 00 08 00 10 86 BF C6 EB 03 06 A4 19 E4 18 01 3F E4 B2 DE B2 DB ……
//Module responds with the received FW packet.
05 5B 07 00 00 21 00 00 20 00 08
……
//Host sends FW packet (*bootloader.bin* 4096 bytes). (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0C 10 00 21 00 60 00 08 00 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ……
//Module responds with the received FW packet.
05 5B 07 00 00 21 00 00 60 00 08
//Host sends FW packet (*LC76GABNRxxAxxS.bin* 4096 bytes). (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0C 10 00 21 00 90 00 08 00 10 DF F8 C8 D0 72 B6 32 48 80 47 32 48 32 49 01 60 32 ……
//Module responds with the received FW packet.
05 5B 07 00 00 21 00 00 90 00 08
……
//Host sends FW packet (*LC76GABNRxxAxxS.bin* 4096 bytes). (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0C 10 00 21 00 80 10 08 00 10 93 42 02 D3 0E 48 DB 21 DC E7 D3 1A 4B 45 01 93 3C ……
//Module responds with the received FW packet.
05 5B 07 00 00 21 00 00 80 10 08
//Host sends FW packet (*gnss_config.bin* 4096 bytes). (*Sample Code for FW Packet and Format Address Checksum*)
05 5A 0C 10 00 21 00 70 1F 08 00 10 01 01 01 01 FF FF FF FF FF FF FF FF FF FF FF FF 02 01 ……
//Module responds with the received FW packet.
05 5B 07 00 00 21 00 00 70 1F 08
//*Figure 12: Flow of Auto Reboot*
//Host sends a reboot command to the module.
05 5A 03 00 06 21 01
//Module reboots after responding to the command.
05 5B 03 00 06 21 00

4 Appendix A References

Table 13: Related Document

Document Name
[1] Quectel_LC86G_Series_EVB_User_Guide
[2] Quectel_LC76G_Series_EVB_User_Guide
[3] Quectel_LC26G(AB)_EVB_User_Guide
[4] Quectel_LC26G(AB)&LC76G&LC86G_Series_GNSS_Protocol_Specification

Table 14: Terms and Abbreviations

Abbreviation	Description
BROM	Boot ROM
CRC	Cyclic Redundancy Check
DA	Download Agent
FW	Firmware
GNSS	Global Navigation Satellite System
UART	Universal Asynchronous Receiver/Transmitter
WDT	Watchdog Timer