GigaDevice Semiconductor Inc.

MCU IN-SYSTEM PROGRAMMING PROTOCOLS

BASED ON USART

Table of Contents

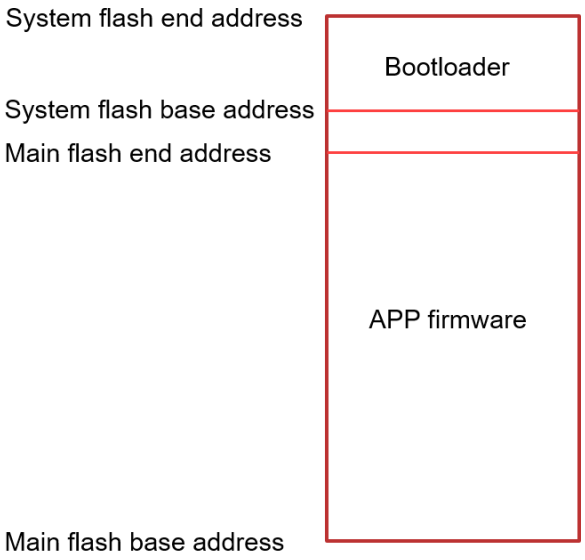
List of Figures

List of Tables

1 General description

This manual describes the in-system programming (ISP) protocol used by the Giga Device (GD) MCU. ISP is a technique where a programmable device is programmed after the device is placed in a circuit board. ISP avoids a separate programming stage prior to assembling the system. Commonly used ISP protocols are based on USART, USB, I2C, etc.

Figure 1-1. ISP bootloader and the application firmware in the MCU flash memory



For USART and USB ISP, a bootloder is needed, and it is programmed in the system flash memory by GD when the chip is manufactured and can't be modified by the user.

All the MCUs support USART ISP.

The steps to setup the USART ISP communication:

- Connect Boot0 pin to VCC, Connect Boot1 pin to GND
- Connect the USART pins to PC
- Reset the MCU to boot from system memory
- Open the GigaDevice ISP Programmer and Click "Next"

Table 1-1. MCU Memory region available for the ISP

PRODUCT	RAM	MAIN FLASH	ERASE/PROGRAM PROTECTION UNIT	OPTION BYTES	SYSTEM FLASH
GD32F303VET6	0x20000200-0x2000FFFF	0x08000000-0x0807FFFF	4KB	0x1FFFF800-0x1FFFF80F	0x1FFFF000-0x1FFFF7FF
GD32E230C8T6	0x20000F00-0x20001FFF	0x08000000-0x0800FFFF	1KB	0x1FFFF800-0x1FFFF80B	0x1FFFEC00-0x1FFFF7FF
GD32F450	0x20003000-0x2006FFFF	0x08000000-0x083BFFFF	Sector	0x1FFFC000-0x1FFFC00F, 0x1FFEC000-0x1FFEC00F	0x1FFF0000-0x1FFF7A2F

Table 1-2. MCU USART pin pairs available for the ISP

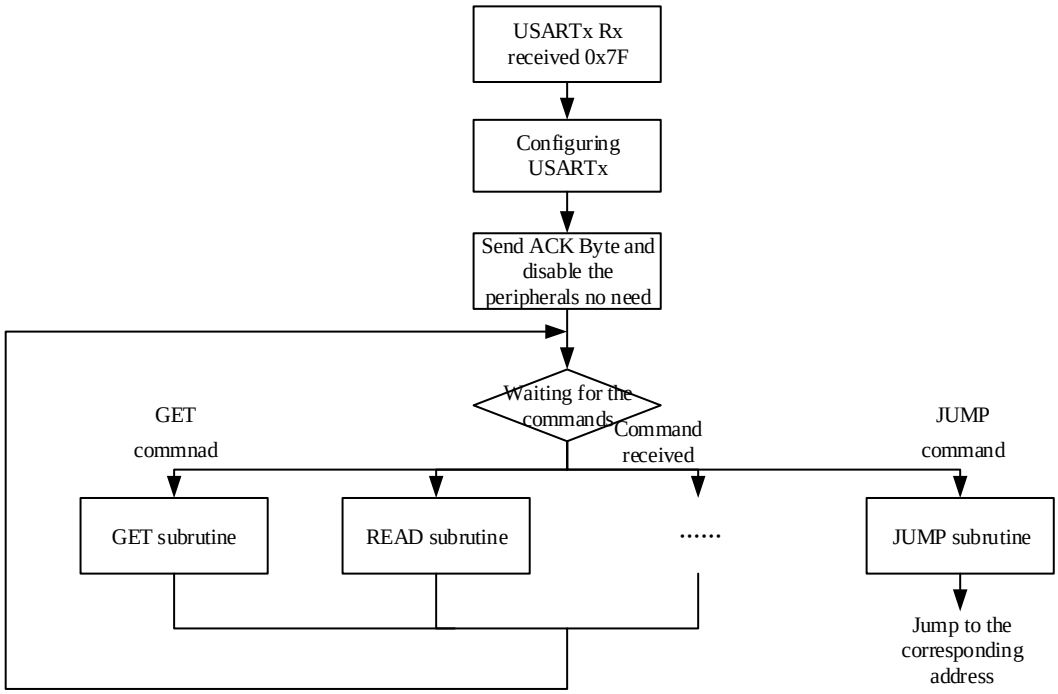
PRODUCT	Pair 1	Pair 2	Pair 3
GD32F303VET6	PA9/PA10	-	-
GD32E230C8T6	PA9/PA10	PA2/PA3	PA14/PA15
GD32F450	PA9/PA10	PB10/PB11	PC10/PC11

Notes:

- The bootloader won't boot successfully if the PLL is not locked when the power is not stable.
- For the bootloader supporting several USART pin pairs, if one pair is used in USART ISP, the signals in other pairs should be stable at the beginning.

2 ISP BOOTLOADER FLOW CHART

Figure 2-1. ISP bootloader flow chart



When the bootloader boots, the bootloader begins to detect the **handshake byte (0x7F)** from the Rx pin of the corresponding USARTx. The signal frame of the handshake byte includes one start bit, one stop bit, **even parity check enabled** with 0x7F as the data bits.

Using the systick timer to calculate the interval of the two rising edges of the handshake byte frame. The baudrate detected is calculated with the timer interval value and the frequency of the system clock.

Then the bootloader will initialize the specific USARTx, from whose Rx pin the handshake byte is received. The bootloader replys to the PC with **acknowledge (ACK) byte (0x79)**, using the baudrate just calculated.

Note: In case of the systick timer overflow, the baudrate used should be larger than 1200 and lower if 115200 fails.

3 BOOTLOADER COMMANDS SUPPORTED

Table 3-1. bootloader commands supported

Command	CODE	description
GET*	0x00	Get the version of the bootloader and the commands supported
GET VERSION*	0x01	Get the version of the bootloader
GET GROUP*	0x02	Get the group which the chip belongs to
READ	0x11	Read no more than 256 bytes data from specified address
JUMP	0x21	Jump to an address to execute the codes there
PROGRAM	0x31	Program no more than 256 bytes data from a specified address
ERASE	0x43	Erase one or more pages of the main flash
EXTEND ERASE	0x44	Erase one or more pages of the main flash using 2 bytes addressing information
ENABLE ERASE/ PROGRAM PROTECTION	0x63	Enable some pages under the protection from erasing or programming
DISABLE ERASE/ PROGRAM PROTECTION	0x73	Disable the protection from erasing or programming on the whole main flash
ENABLE SECURITY PROTECTION	0x82	Enable the security protection
DISABLE SECURITY PROTECTION*	0x92	Disable the security protection
FURTHER USE	0x06	RESERVED

When the security protection status is enabled, only the commands marked with * are supported. Other commands will be replied with a **non-acknowledge (NACK) byte (0x1F)**. The GET VERSION command and FURTHER USE command are only supported by some of the GD MCU. The ERASE command and EXTEND ERASE command are exclusive in a particulier bootloader. The user should send the GET command first to check which commands are supported.

The bootloader will response a NACK byte, when the received CODE is not one listed in the above table.

To verify the data communication between the MCU and PC.

- CHECKSUM: if the data is more than one byte, then the transmitter will calculate the XOR value of the data byte flow as the last sending byte, the CHECKSUM. While the receiver will calculate the XOR value of the data byte flow and the CHECKSUM received. If the transmission is correct, the XOR value calculated by the receiver will be 0x00.

- Complement Byte: if the data is only one byte, then the transmitter will also send the complement byte of the data to the receiver, and the receiver will calculate the XOR value of the two bytes.

4 GET COMMAND

Figure 4-1. GET command subroutine flow (PC)

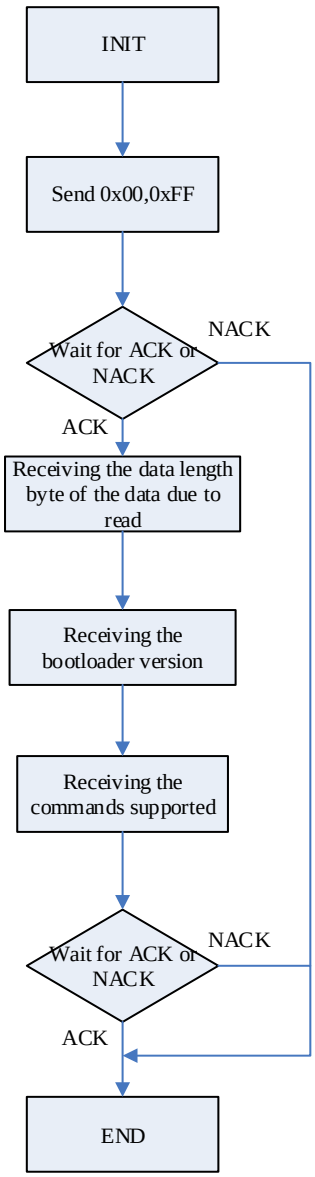
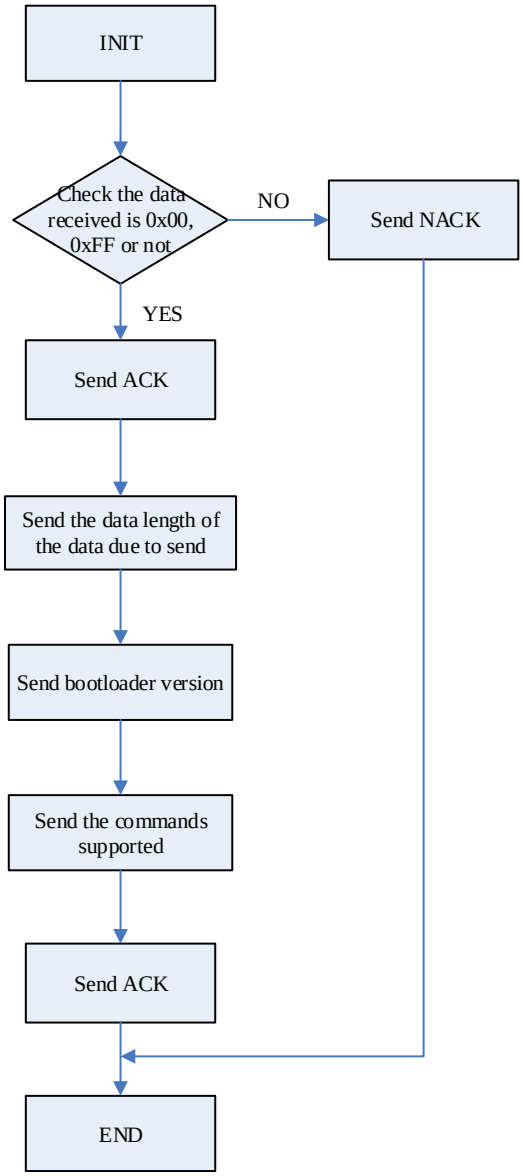


Figure 4-2. GET command subroutine flow (bootloader)



The bootloader will send the data as follows:

FIRST BYTE: ACK

2nd BYTE: 0x0B (the data length due to send minus 1, 11 commands and the bootloader version byte due to send)

3rd BYTE: bootloader version

4th BYTE: 0x00

5th BYTE: 0x01

6th BYTE: 0x02

7th BYTE: 0x11

8th BYTE: 0x21

9th BYTE: 0x31

10th BYTE: 0x43

11th BYTE: 0x63

12th BYTE: 0x73

13th BYTE: 0x82

14th BYTE: 0x92

15th BYTE: ACK

5 GET VERSION COMMAND

Figure 5-1. GET VERSION command subroutine flow (PC)

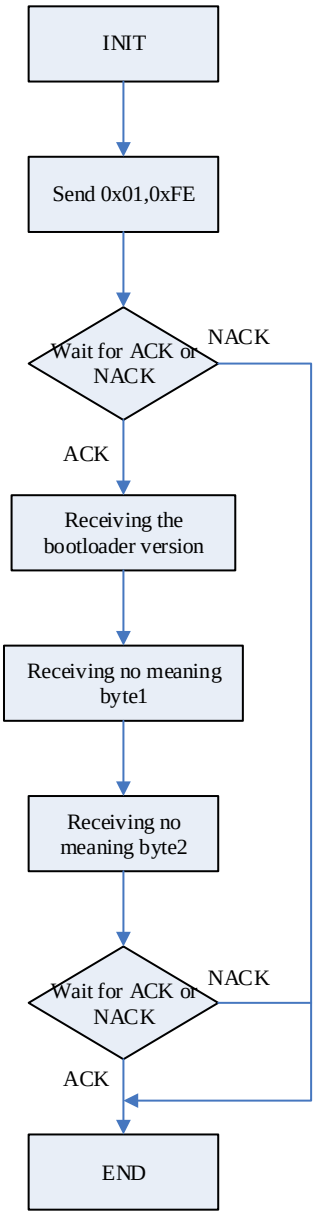
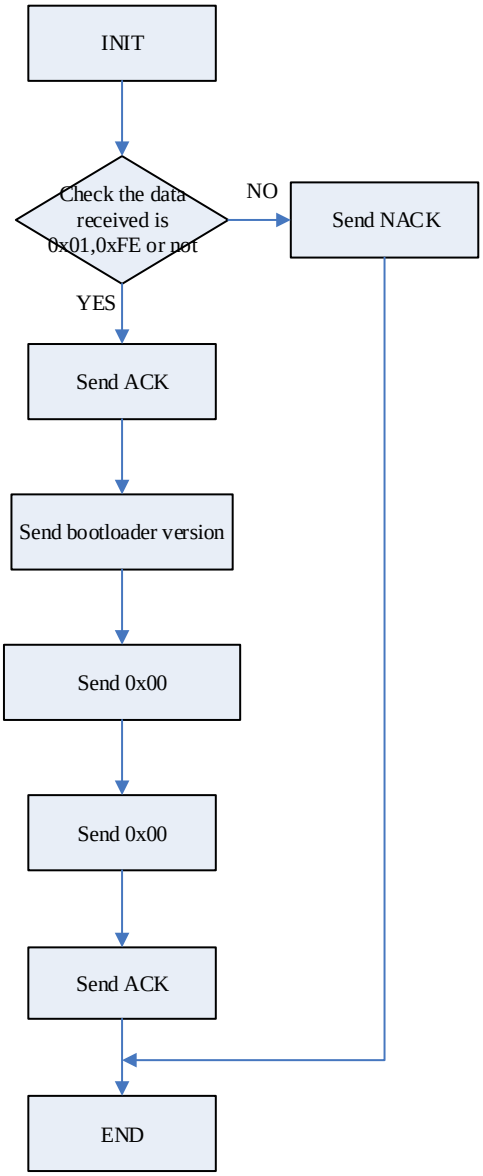


Figure 5-2. GET VERSION command subroutine flow (bootloader)



6 GET GROUP COMMAND

Figure 6-1. GET GROUP command subroutine flow (PC)

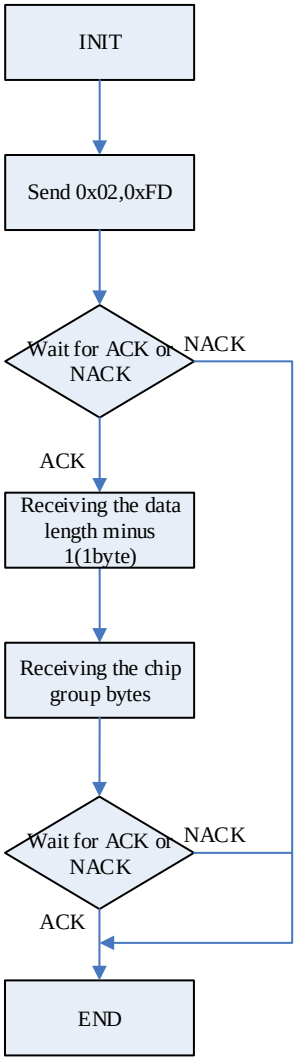
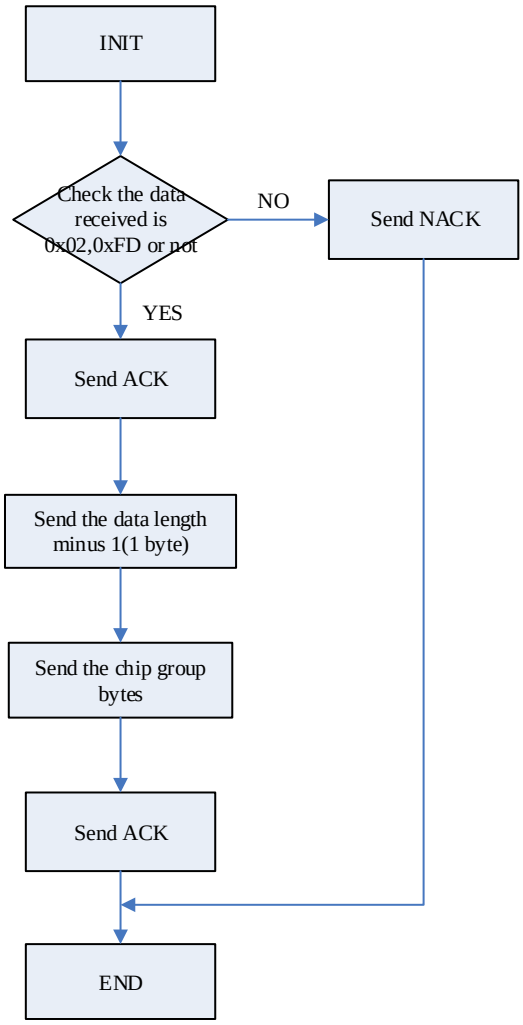


Figure 6-2. GET GROUP command subroutine flow (bootloader)



The bootloader will send the data as follows:

FIRST BYTE: ACK

2nd BYTE: 0x01 (the data length due to send minus 1, the group information include 2 bytes)

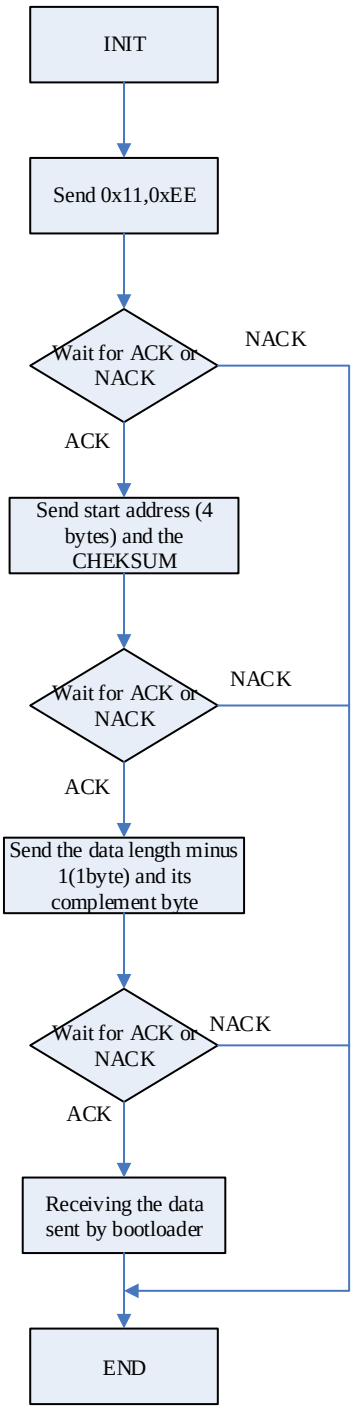
3rd BYTE: GROUP HIGH BYTE

4th BYTE: GROUP LOW BYTE

5th BYTE: ACK

7 READ COMMAND

Figure 7-1. READ command subroutine flow (PC)



The PC send the byte flow to the MCU as follows:

FIRST BYTE: 0x11

2nd BYTE: 0xEE

Wait for ACK

3rd BYTE: START ADDRESS (HIGH BYTE)

4th BYTE: START ADDRESS (MEDIUM HIGH BYTE)

5th BYTE: START ADDRESS (MEDIUM LOW BYTE)

6th BYTE: START ADDRESS (LOW BYTE)

7th BYTE: CHECKSUM of the START ADDRESS

Wait for ACK

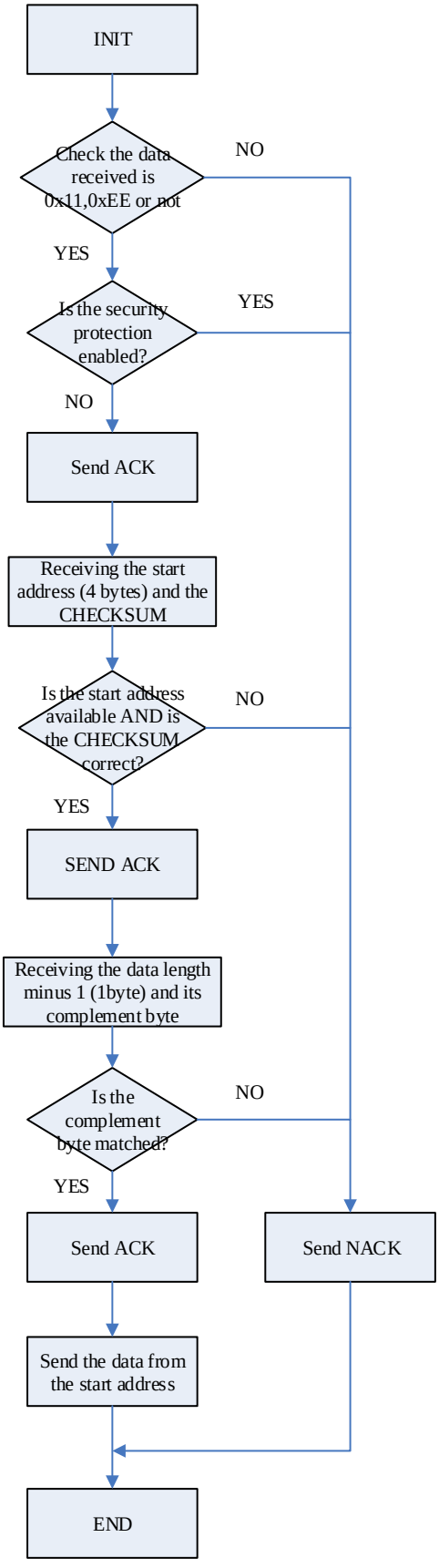
8th BYTE: the length of the data due to read minus 1 (0<=N<=255)

9th BYTE: the complement byte of the 8th BYTE

Wait for ACK

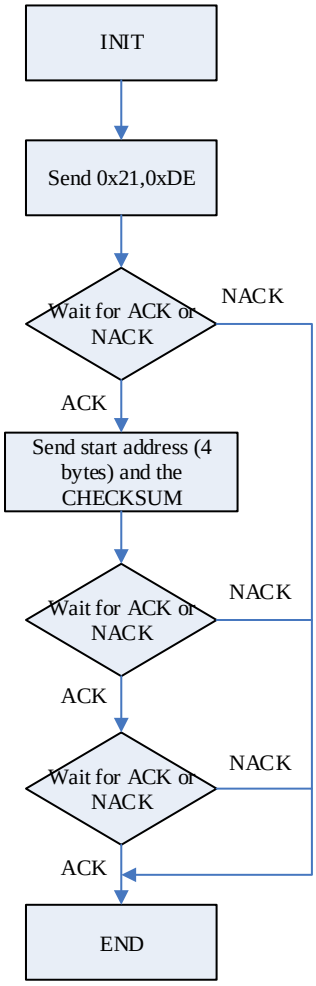
Receiving the data flow (LSB first)

Figure 7-2. READ command subroutine flow (bootloader)



8 JUMP COMMAND

Figure 8-1. JUMP command subroutine flow (PC)



Note: If the bootloader jump command response is implemented following subroutine flow 1 or subroutine flow 2, only one ACK will be received after sending the start address.

The PC send the byte flow to the MCU as follows:

FIRST BYTE: 0x21

2nd BYTE: 0xDE

Wait for ACK

3rd BYTE: START ADDRESS (HIGH BYTE)

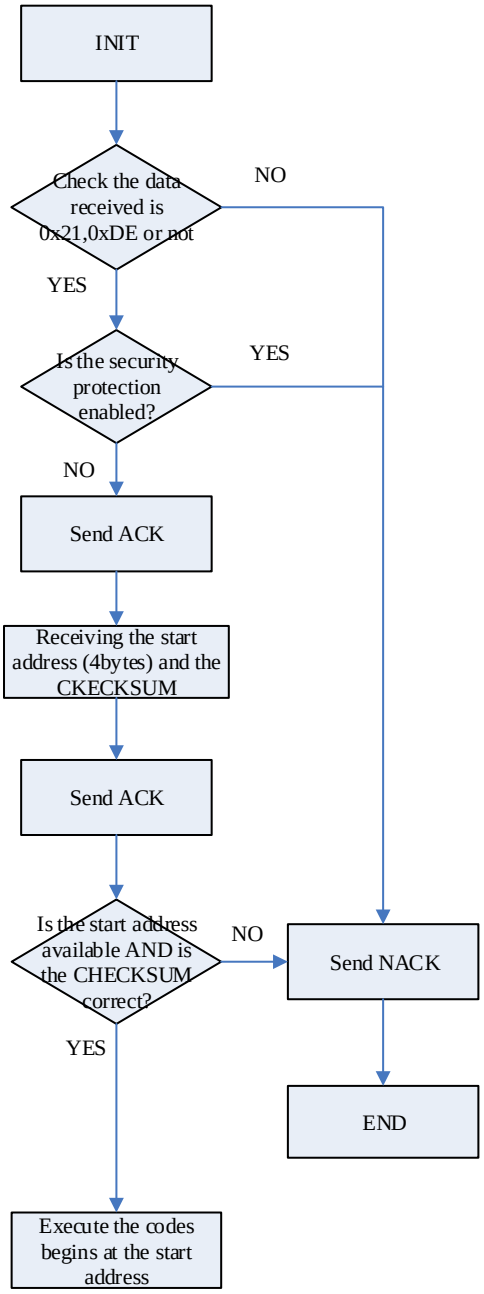
4th BYTE: START ADDRESS (MEDIUM HIGH BYTE)

5th BYTE: START ADDRESS (MEDIUM LOW BYTE)

6th BYTE: START ADDRESS (LOW BYTE)

7th BYTE: CHECKSUM of the START ADDRESS

Figure 8-2. JUMP command subroutine flow 1 (bootloader)



To execute the codes begins at the start address, the bootloader acts as follows:

- Change the MSP value with the value stored at the start address
- The cortex core fetches the instruction stored at the start address + 4 (reset handler)
- If the bootloader does not support FURTHER USE command, the jump command response is implemented following subroutine flow 1. If the MCU is GD32F4xx series, the jump command response is implemented following subroutine flow 2. Otherwise, it follows subroutine flow 3.

Figure 8-3. JUMP command subroutine flow 2 (bootloader)

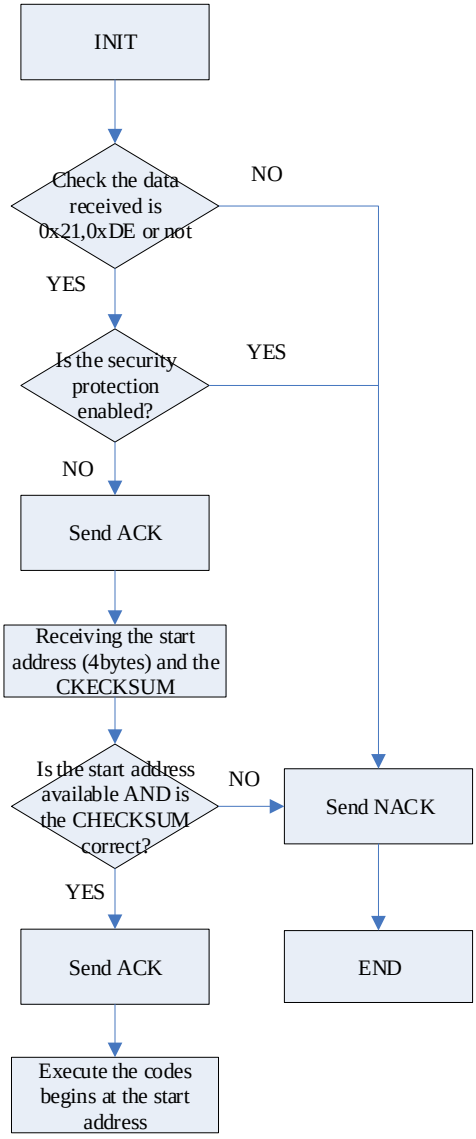
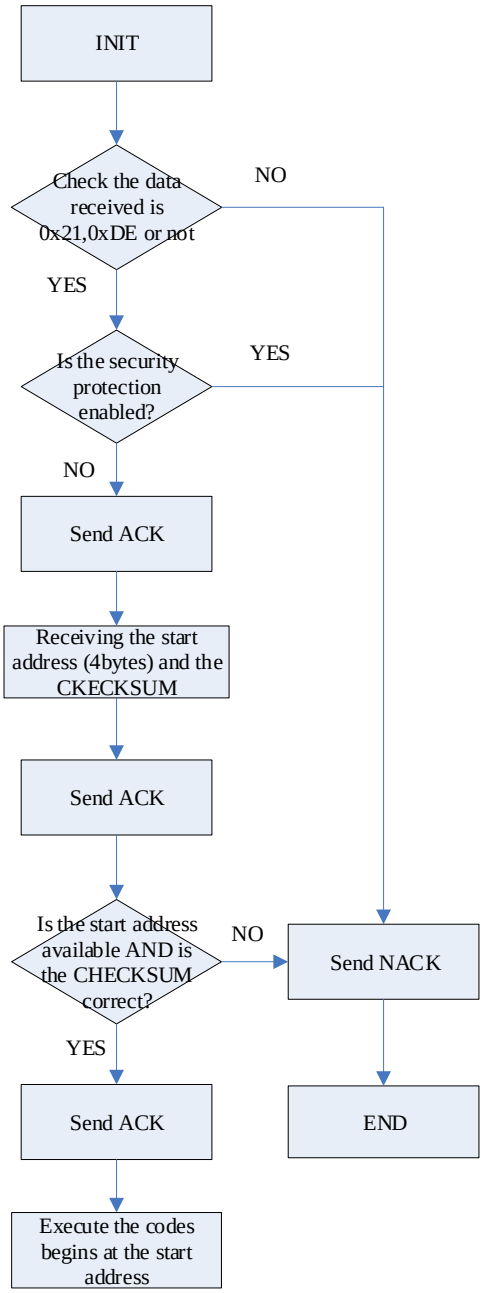
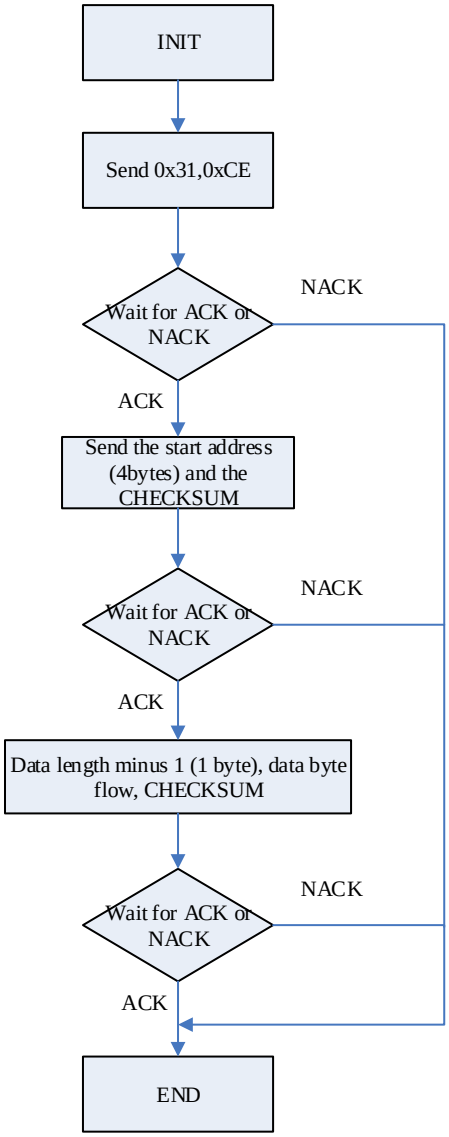


Figure 8-4. JUMP command subroutine flow 3 (bootloader)



9 PROGRAM COMMAND

Figure 9-1. PROGRAM command subroutine flow (PC)



The PC send the byte flow to the MCU as follows:

FIRST BYTE: 0x31

2nd BYTE: 0xCE

Wait for ACK

3rd BYTE: START ADDRESS (HIGH BYTE)

4th BYTE: START ADDRESS (MEDIUM HIGH BYTE)

5th BYTE: START ADDRESS (MEDIUM LOW BYTE)

6th BYTE: START ADDRESS (LOW BYTE)

7th BYTE: CHECKSUM of the START ADDRESS

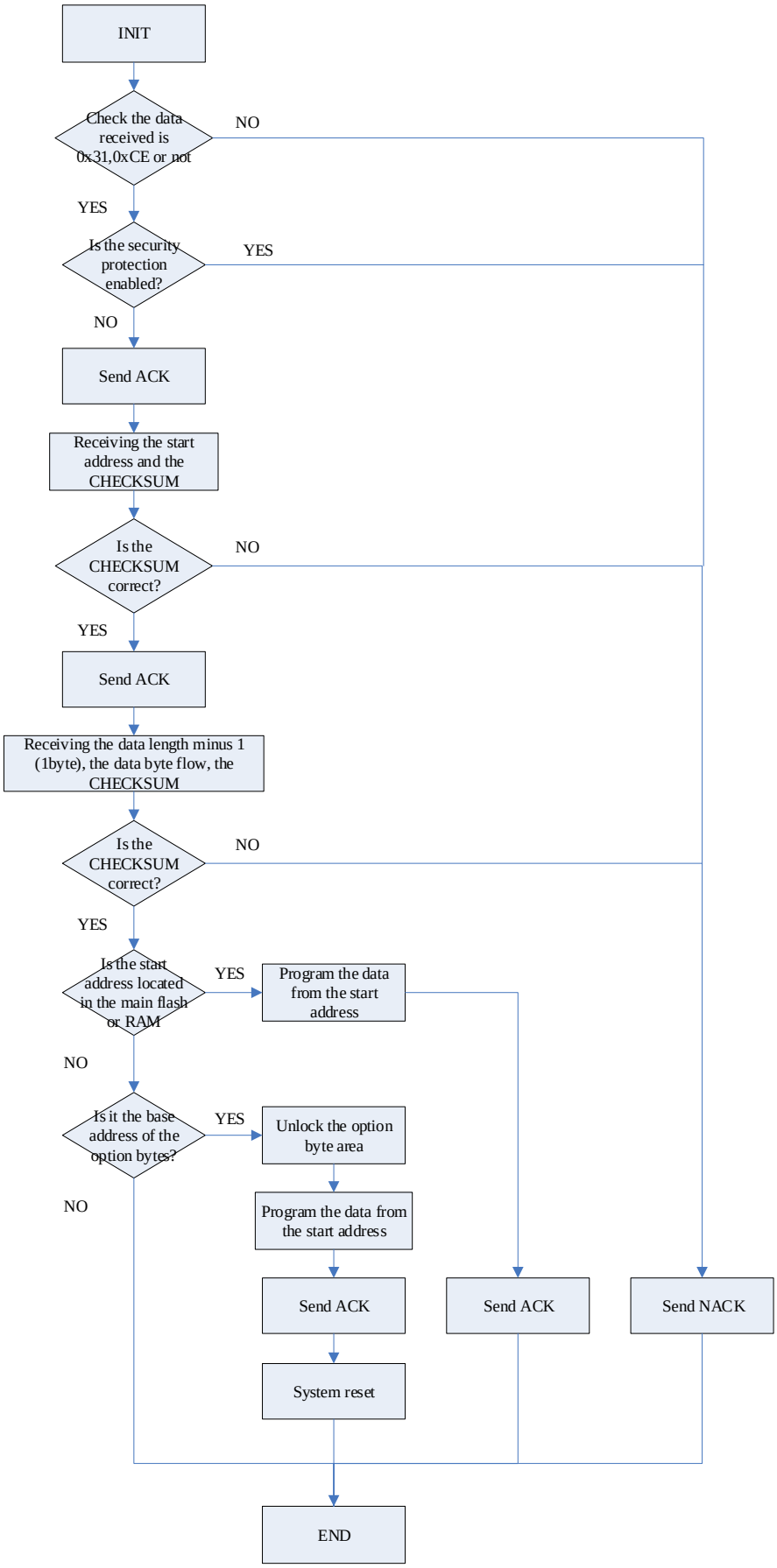
Wait for ACK

8th BYTE: the length of the data due to program minus 1 (0<=N<=255)

DATA FLOW: N+1 bytes (LSB first)

CHECKSUM: XOR value of the 8th BYTE and the DATA FLOW

Figure 9-2. PROGRAM command subroutine flow (bootloader)

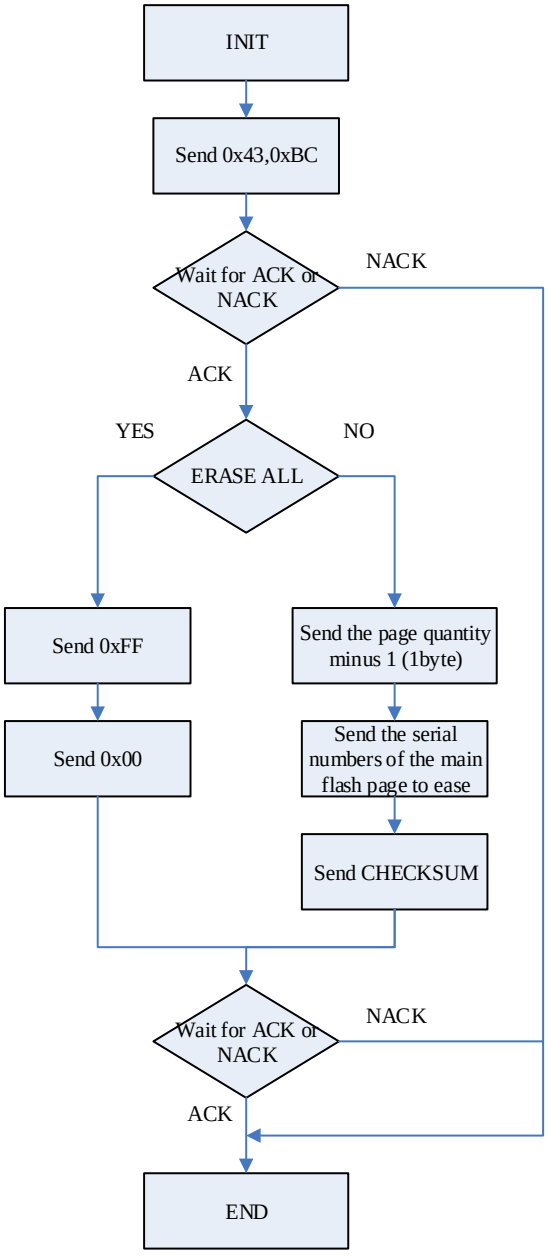


Notes:

- The start address should be word (32-bit) aligned.
- If the program area is in the main flash memory or RAM, the length of the data due to program should be the times of 4. If not, during the last program operation, the bootloader will form a 4-byte word first by combining several redundancy bytes with the original data.
- If the program area is under erase/program protection, the program function will fail but there is no NACK response sent by bootloader.
- If the start address is the base address of the option bytes, the whole option bytes area should be erased before the new value programed.

10 ERASE COMMAND

Figure 10-1. ERASE command subroutine flow (PC)



The PC send the byte flow to the MCU as follows:

FIRST BYTE: 0x43

2nd BYTE: 0xBC

Wait for ACK

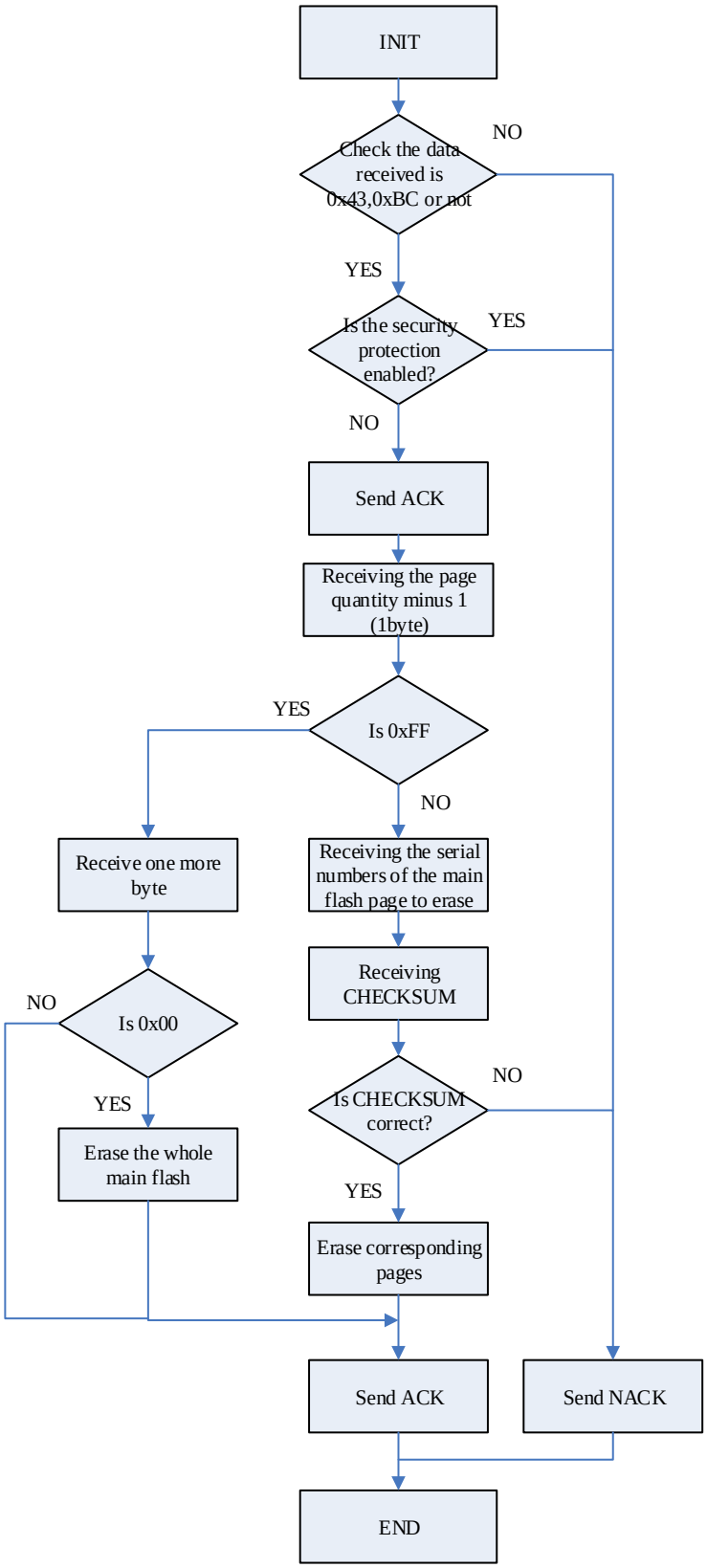
3rd BYTE: N (if N=255=0xFF, that means erase the whole main flash; if 0<=N<=254, that means the quantity of the pages to erase minus 1)

4th BYTE: 0x00 (last byte if N=0xFF) or the serial number of the first main flash page to erase (if 0<=N<=254)

DATA FLOW: The remain serial numbers of the main flash pages to erase (N bytes)

CHECKSUM: XOR value of the 3rd BYTE, 4th BYTE and the DATA FLOW

Figure 10-2. ERASE command subroutine flow (bootloader)



Note: If the page to erase is under erase/program protection, the erase function will fail but there is no NACK response sent by bootloader.

11 EXTEND ERASE COMMAND

Figure 11-1. EXTEND ERASE command subroutine flow (PC)

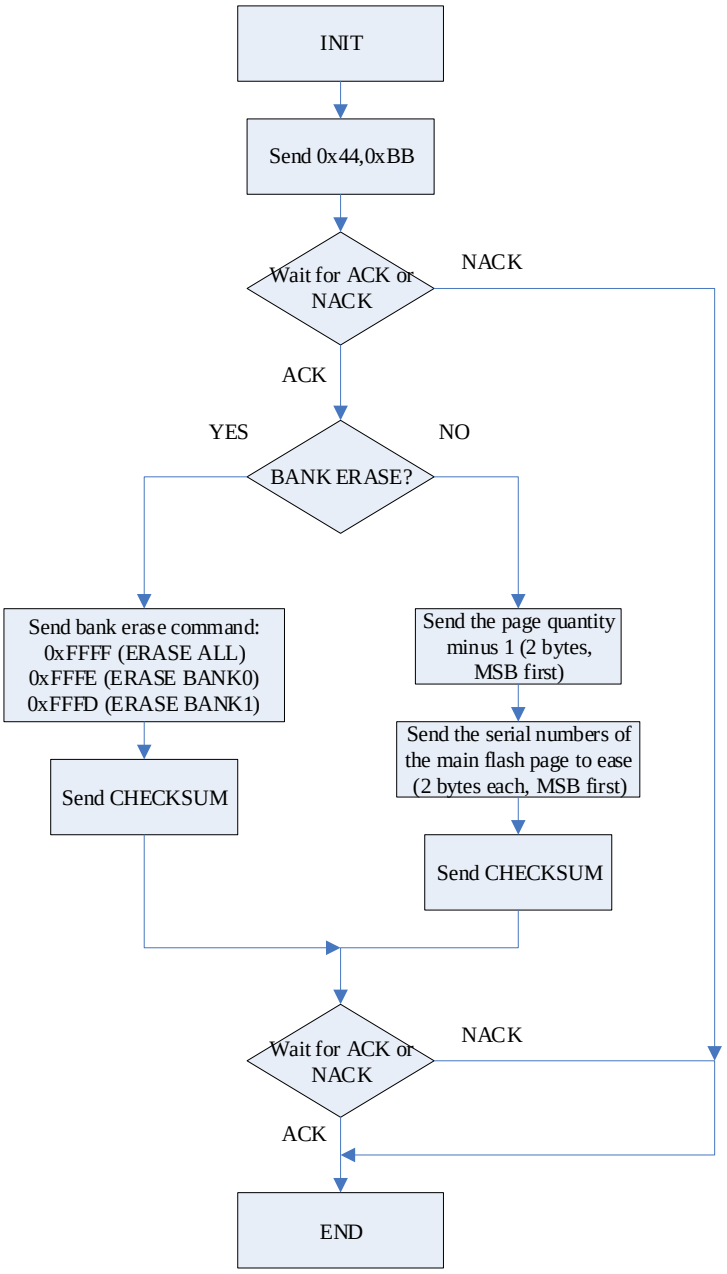
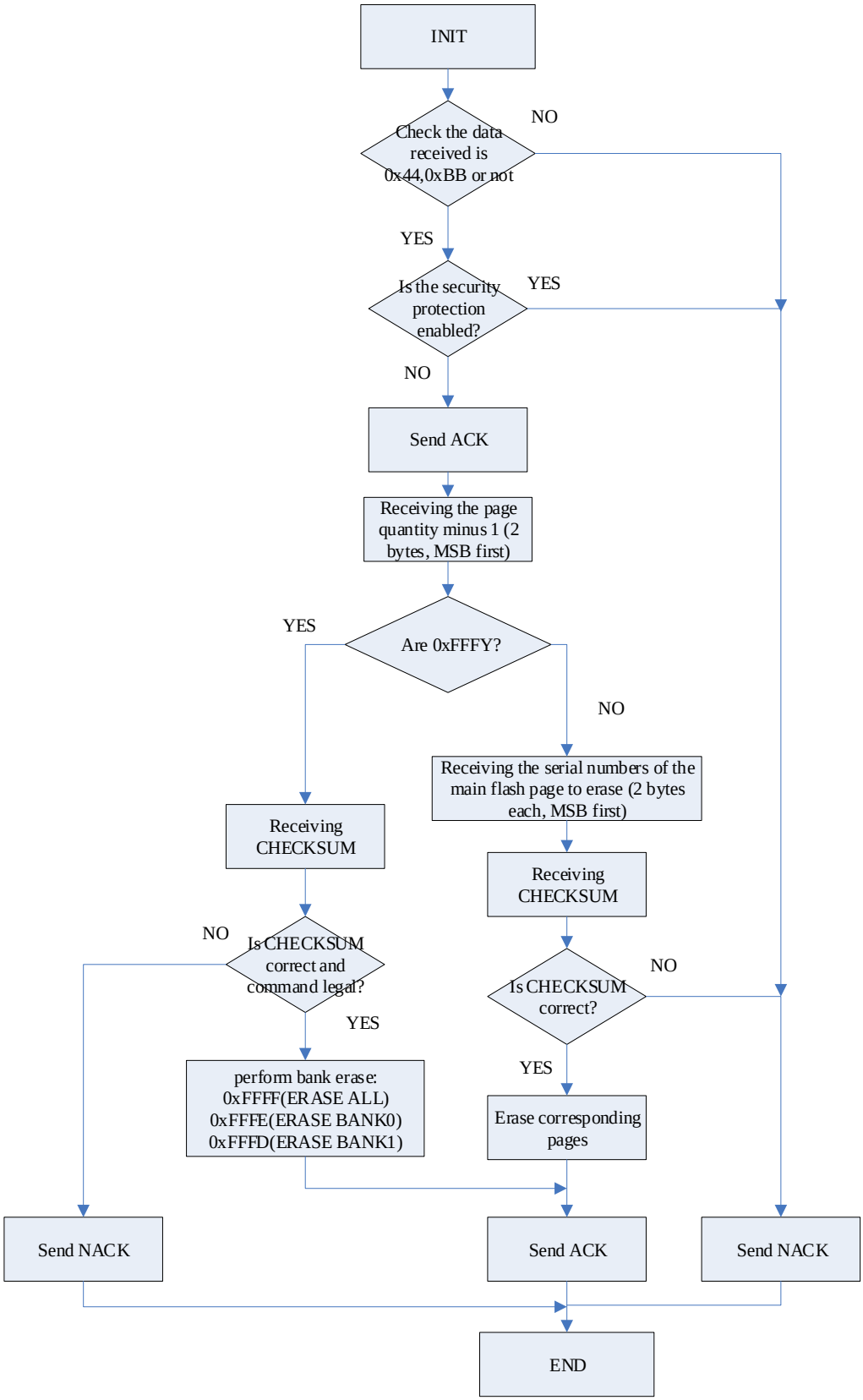
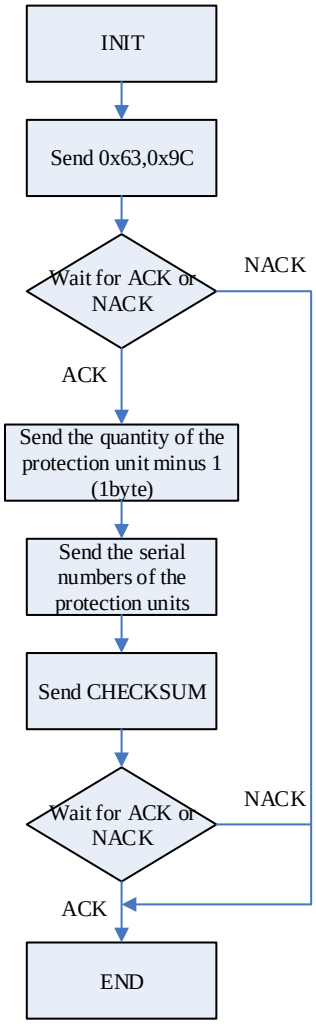


Figure 11-2. EXTEND ERASE command subroutine flow (bootloader)



12 ENABLE ERASE/PROGRAM PROTECTION COMMAND

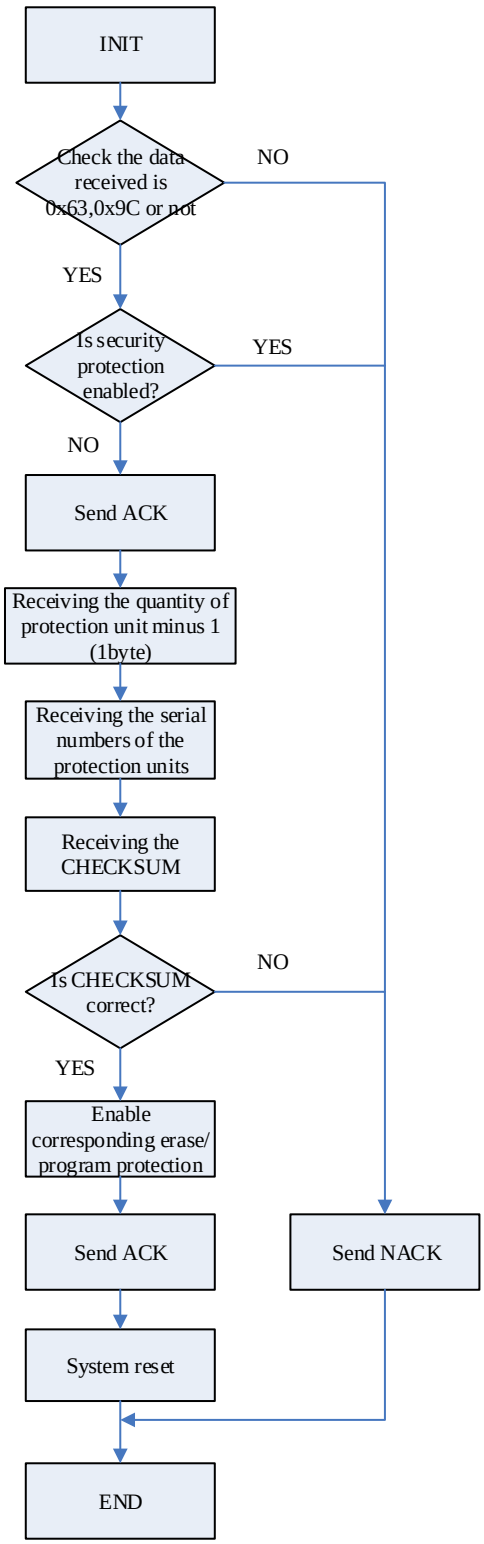
Figure 12-1. ENABLE ERASE/PROGRAM PROTECTION command subroutine flow (PC)



Notes:

- The bootloader won't check if the quantity and serial numbers of the erase/program protection units are matched with the particular MCU product parameters.
- If the bootloader gets a new ENABLE ERASE/PROGRAM PROTECTION command, the main flash units under the erase/program protection which were enabled by previous ENABLE ERASE/PROGRAM PROTECTION command are invalid. Then the main flash units corresponding to the new command are under the protection.

Figure 12-2. ENABLE ERASE/PROGRAM PROTECTION command subroutine flow (bootloader)



13 DISABLE ERASE/PROGRAM PROTECTION COMMAND

Figure 13-1. DISABLE ERASE/PROGRAM PROTECTION command subroutine flow (PC)

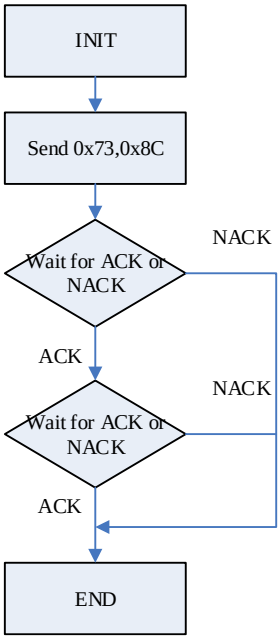
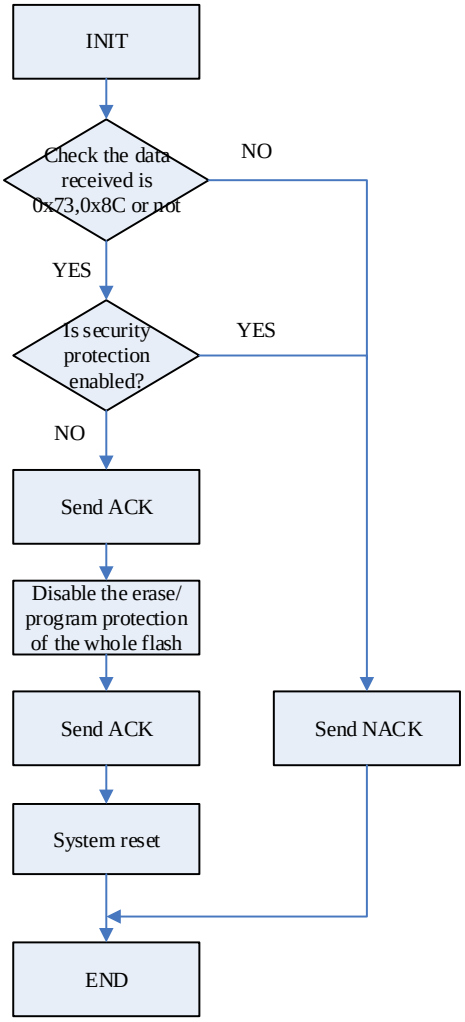


Figure 13-2. DISABLE ERASE/PROGRAM PROTECTION command subroutine flow (bootloader)



14 ENABLE SECURITY PROTECTION COMMAND

Figure 14-1. ENABLE SECURITY PROTECTION command subroutine flow (PC)

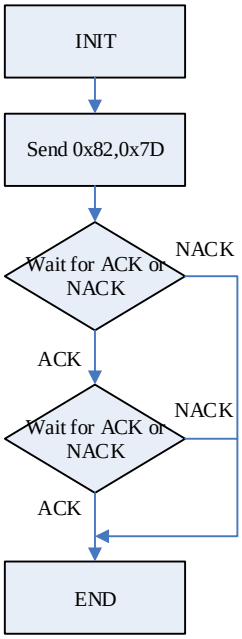
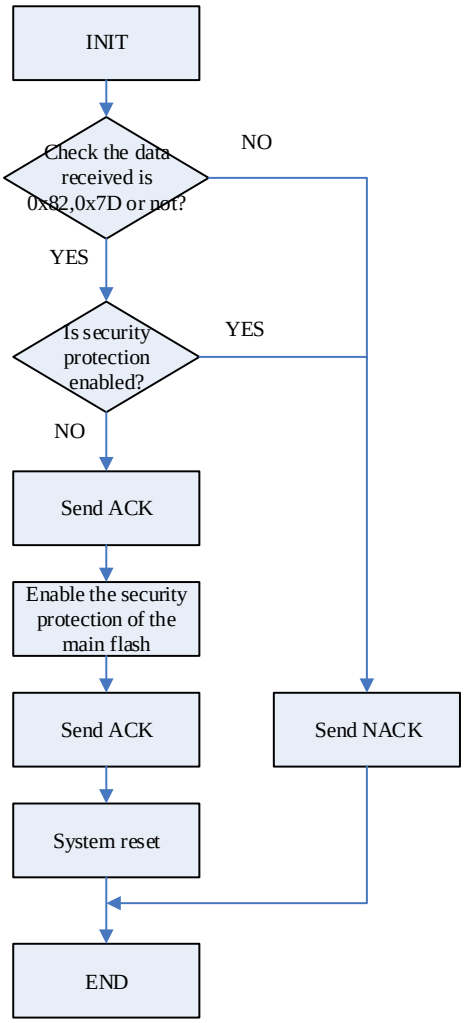


Figure 14-2. ENABLE SECURITY PROTECTION command subroutine flow (bootloader)



15 DISABLE SECURITY PROTECTION COMMAND

Figure 15-1. DISABLE SECURITY PROTECTION command subroutine flow (PC)

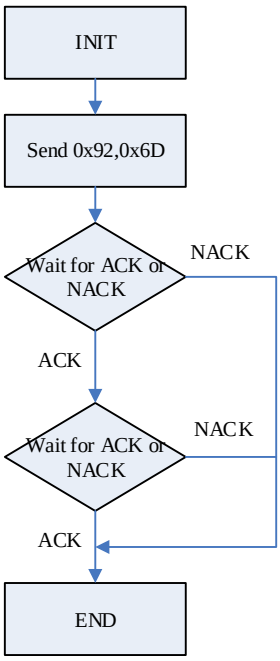
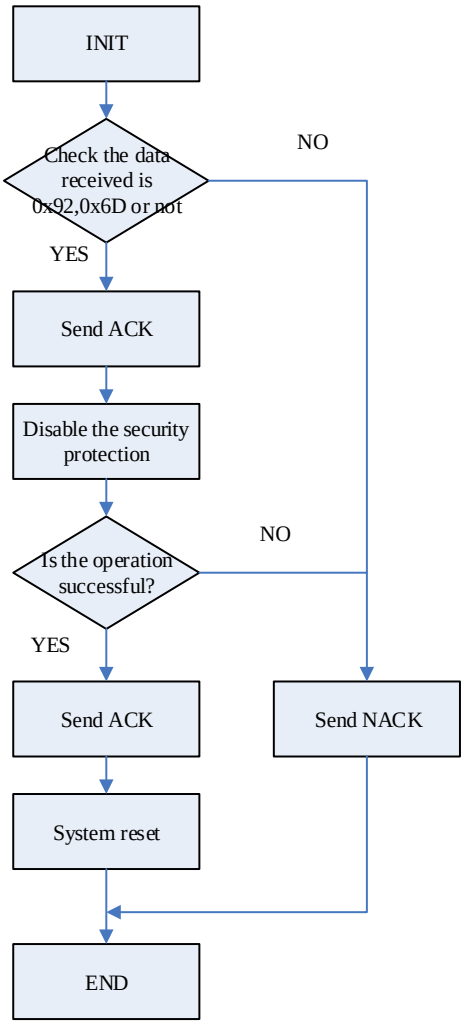


Figure 15-2. DISABLE SECURITY PROTECTION command subroutine flow (bootloader)



16 Revision history

Table 16-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Jun 14, 2019
2.0	Add EXTEND ERASE command chapter, modify JUMP command chapter	Oct 23, 2019

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the “Product”), is owned by the Company under the intellectual property laws and treaties of the People’s Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it’s suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it’s suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.