GigaDevice Semiconductor Inc.

GD ISP PROTOCOL-IIC

Application Note
AN228

Revision 1.0

(Oct. 2024)

Table of Contents

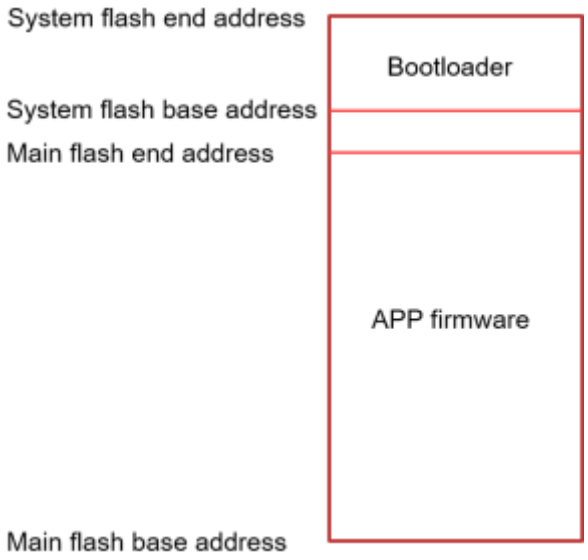
List of Figures

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 I2C.

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



For IIC ISP, 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.

In GD32G5xx series MCU, IIC bootloader is served as a slave device. I2C1 (PC4 and PA8), I2C2 (PC8 and PC9) and I2C3 (PC6 and PC7), and when use I2C port to reprogram, GD-Link v3 should be used.

The steps to setup the IIC ISP communication:

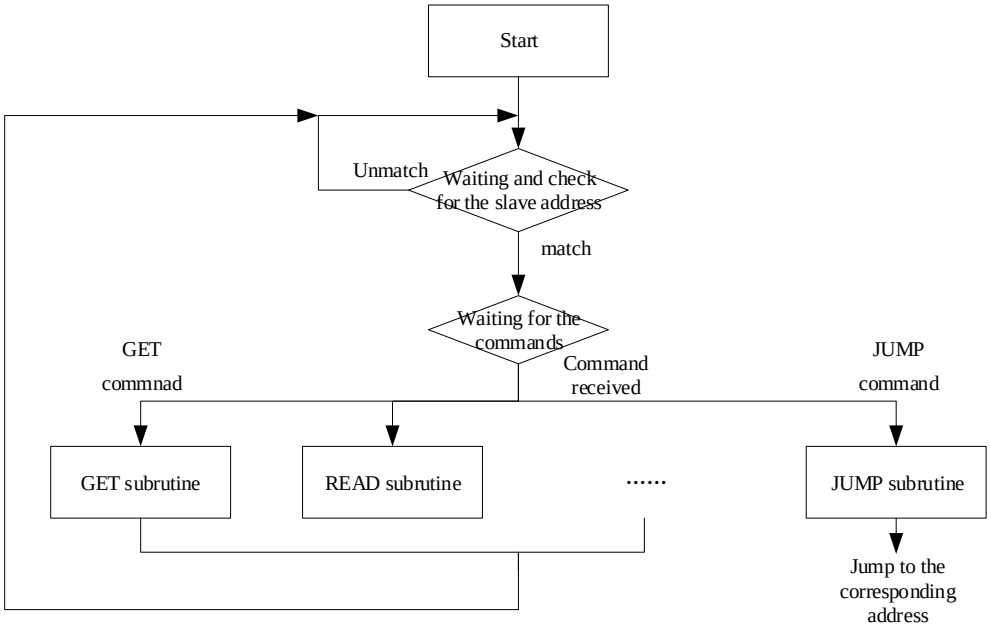
- Connect Boot0 pin to VCC, keep factory values of BOOT_LK, nBOOT1 and nSWBT0
- Connect the IIC pins to a master device
- Reset the MCU to boot from system memory
- Open the GD32 All in One Programmer, "Interface" select "GD ISP Bridge", "Bootloader" select "I2C", and then Click "Connect"

Table 1-1. GD32G5xx Memory region available for the ISP

SRAM	MAIN FLASH	ERASE/PROGRAM PROTECTION UNIT	OPTION BYTES	SYSTEM FLASH	OTP
0x20004000-0x2001FFFF	0x08000000-0x0807FFFF	1KB or 2KB	0x1FFFF800-0x1FFFF82F	0x1FFF0000-0x1FFF33FF	0x1FFF7000-0x1FFF77FF

2 ISP BOOTLOADER FLOW CHART

Figure 2-1. ISP bootloader flow chart



When the code boots from system flash, the bootloader waits for and check the slave address sent from the master. If matches, then bootloader waits for a command. The commands are listed in chapter 3.

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
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	0x44	Erase one or more pages of the main flash
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 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)

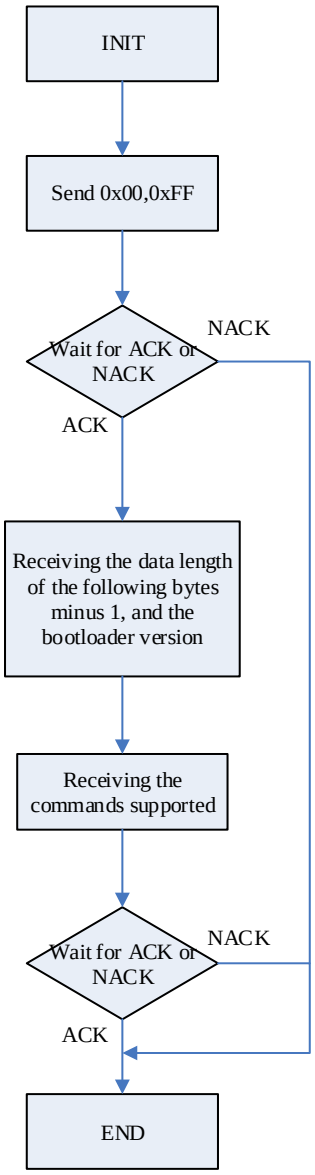
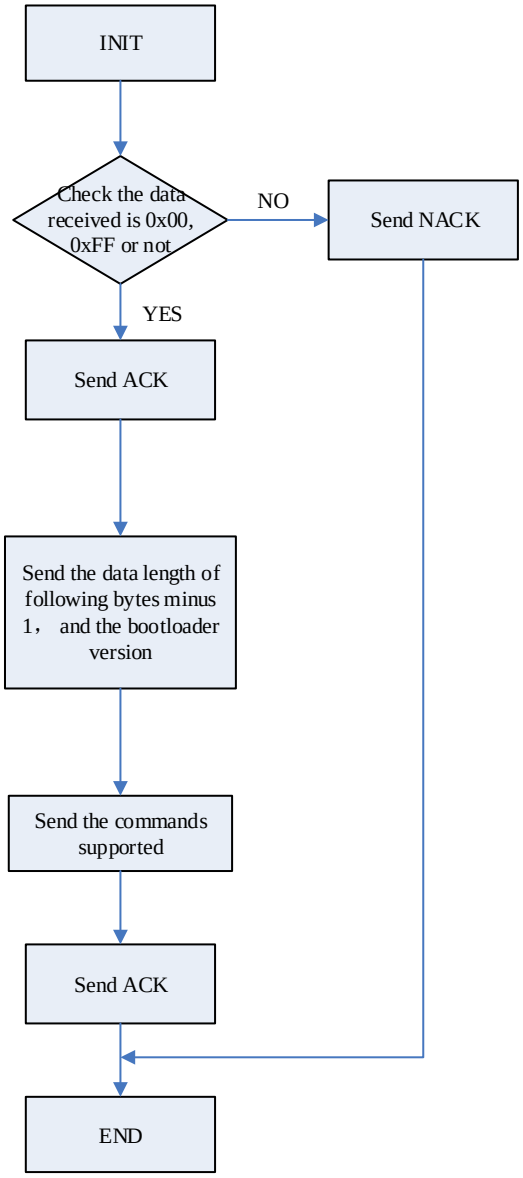


Figure 4-2. GET command subroutine flow (bootloader)



The bootloader will send the data as follows:

FIRST BYTE: ACK

2nd BYTE: 0x0A (the data length due to send minus 1, 10 commands and the bootloader version byte due to send)

3rd BYTE: bootloader version

4th BYTE: 0x00

5th BYTE: 0x11

6th BYTE: 0x21

7th BYTE: 0x31

8th BYTE: 0x44

9th BYTE: 0x63

10th BYTE: 0x73

11th BYTE: 0x82

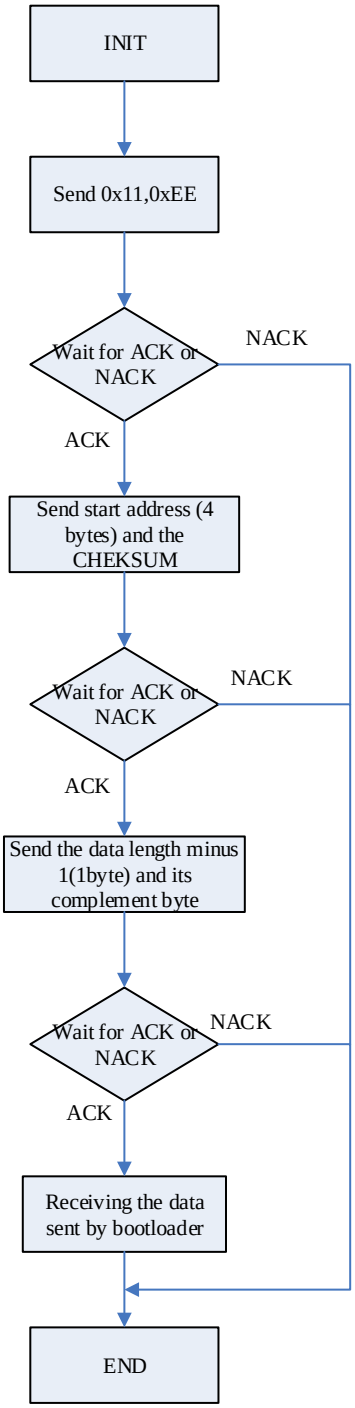
12th BYTE: 0x92

13th BYTE: 0x06

14th BYTE: ACK

5 READ COMMAND

Figure 5-1. 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

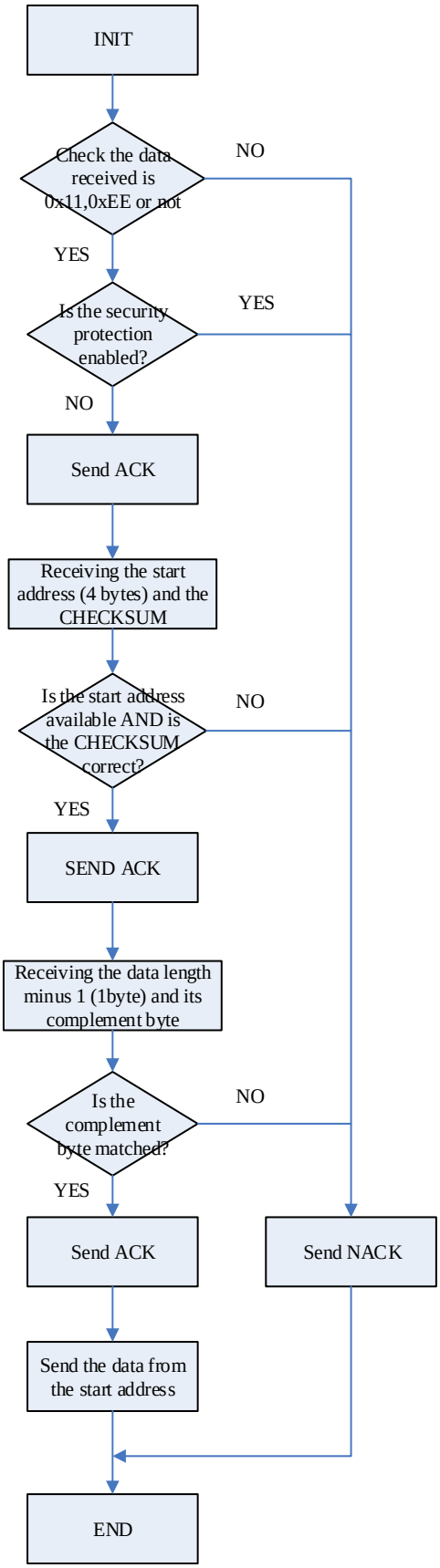
8th BYTE: the length of the data due to read minus 1 (0<=N<=255)

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

Wait for ACK

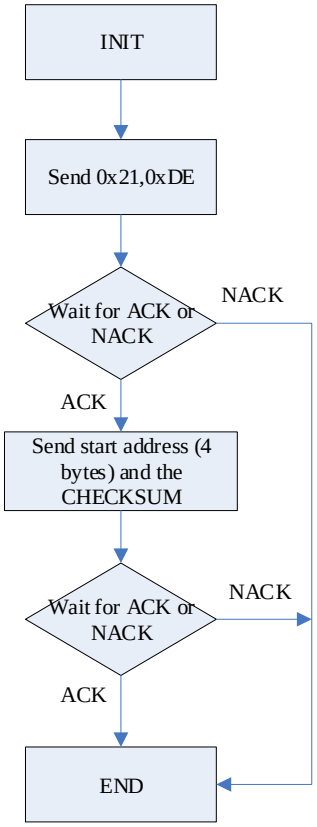
Receiving the data flow (LSB first)

Figure 5-2. READ command subroutine flow (bootloader)



6 JUMP COMMAND

Figure 6-1. JUMP command subroutine flow (PC)



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)

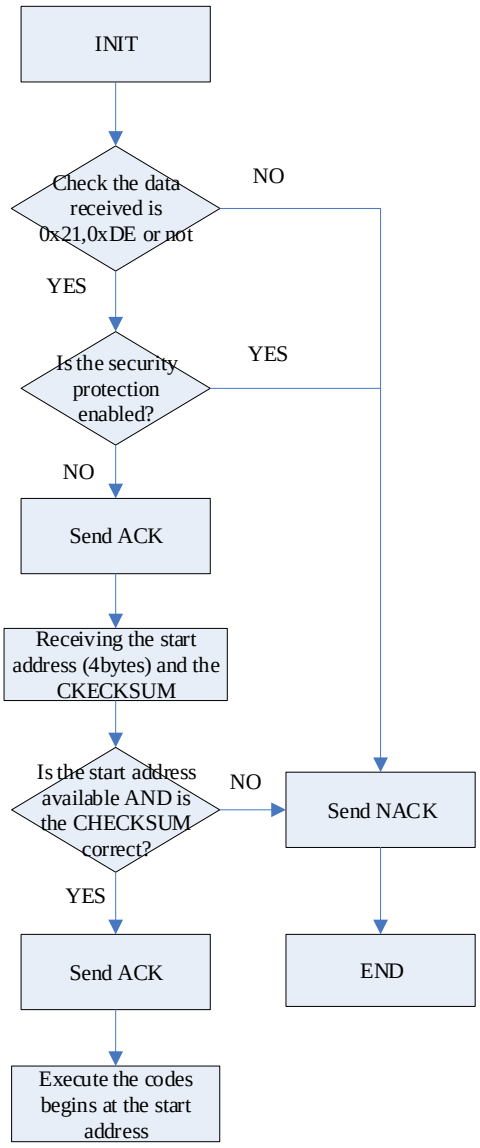
4th BYTE: START ADDRESS (MEDIUM HIGH BYTE)

5th BYTE: START ADDRESS (MEDIUM LOW BYTE)

6th BYTE: START ADDRESS (LOW BYTE)

7th BYTE: CHECKSUM of the START ADDRESS

Figure 6-2. JUMP command subroutine flow (bootloader)

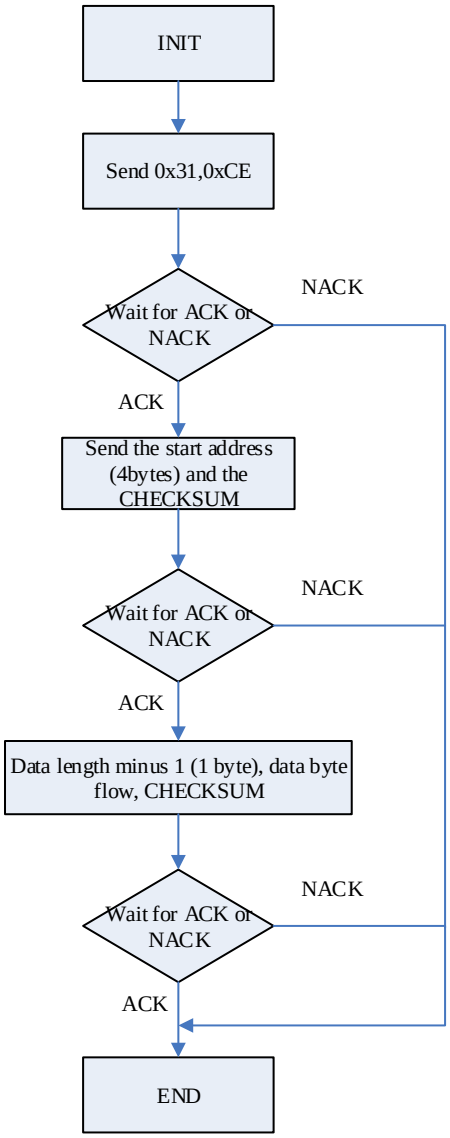


To execute the codes begins at the start address, the bootloader acts as follows:

- 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)

7 PROGRAM COMMAND

Figure 7-1. 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

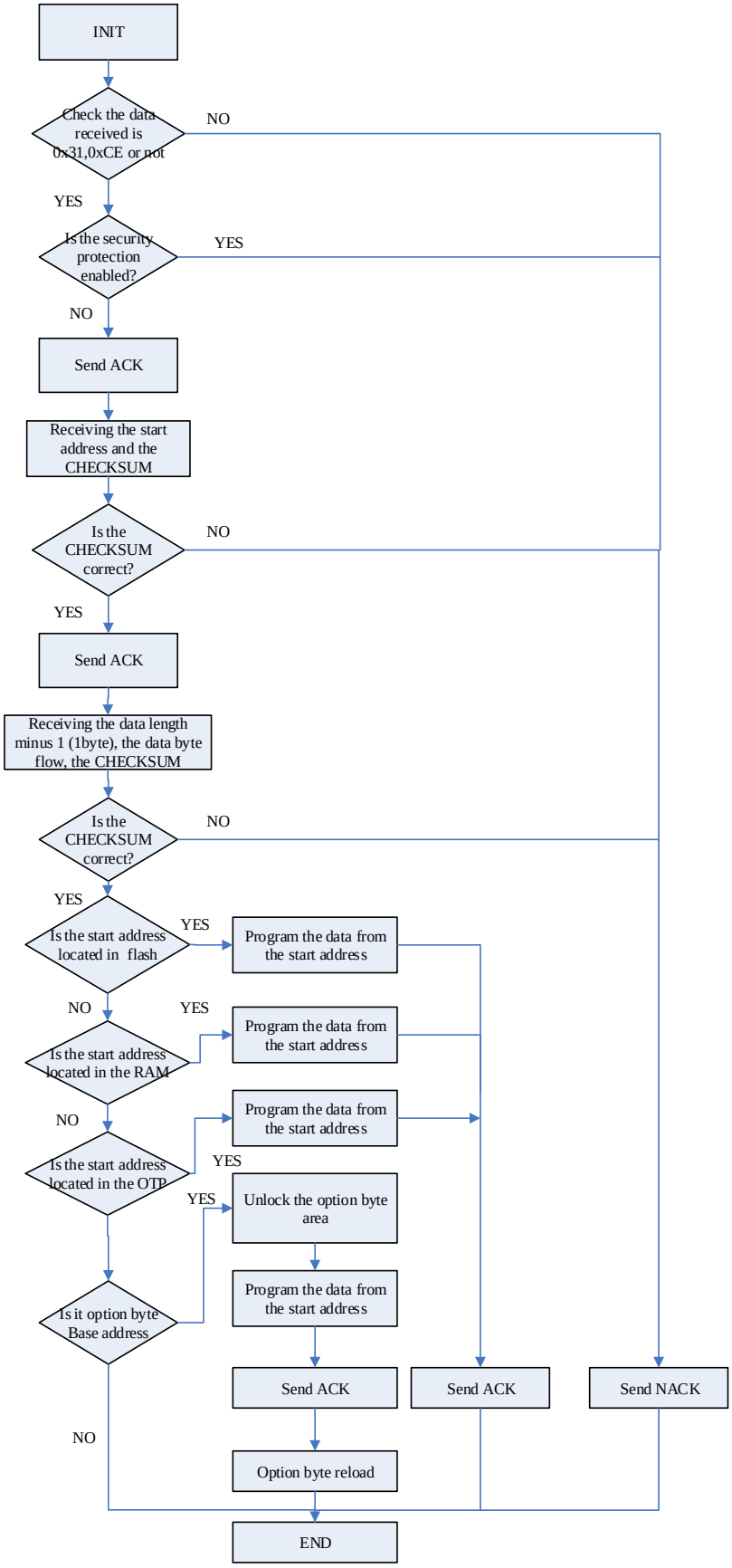
Wait for ACK

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 7-2. PROGRAM command subroutine flow (bootloader)

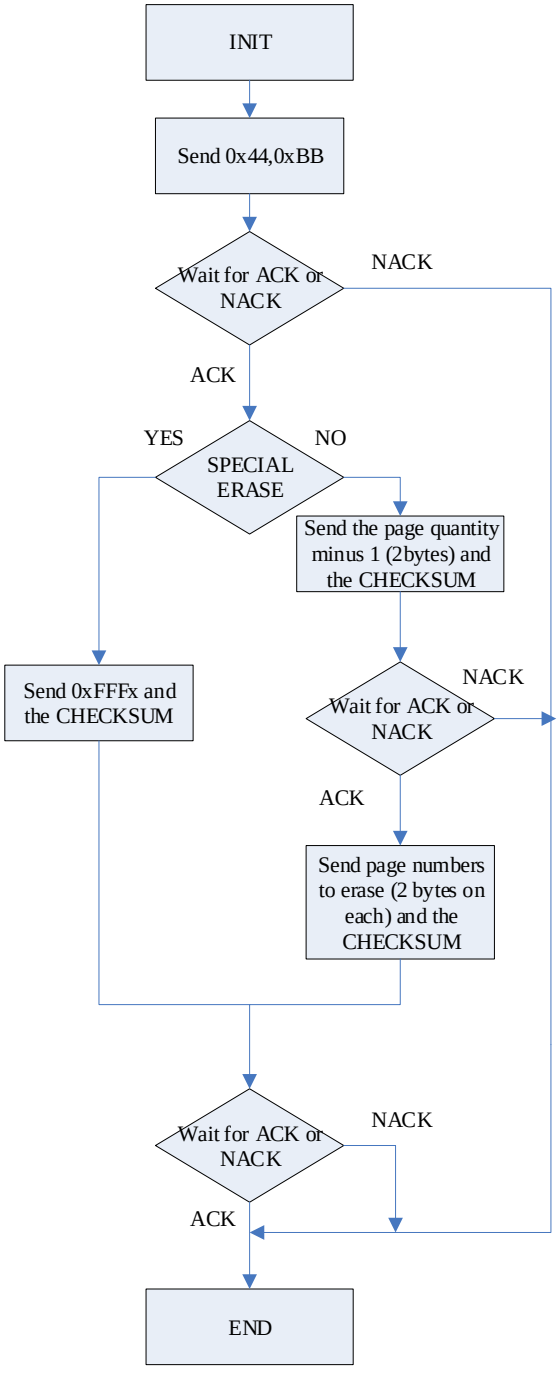


Notes:

- The start address should be double word (64-bit) aligned.
- If the program area is in the main flash memory, OTP or RAM, the length of the data due to program should be the times of 8 respectively. If not, during the last program operation, the bootloader will form 8-byte first by combining several redundancy bytes with the original data.
- If the start address is 0x1FFFF800, the whole option byte area should be erased before the new value programed.

8 ERASE COMMAND

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



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

FIRST BYTE: 0x44

2nd BYTE: 0xBB

Wait for ACK

If special erase:

3rd BYTE: 0xFF

4th BYTE: 0xFx (x can be D, E, F. While D and E means flash bank1 and bank2 erase, F means flash mass erase. In case that GD32G5xx, only F is supported.)

Wait for ACK

If page erase:

3rd and 4th BYTE: Number of pages or sectors to be erased minus 1

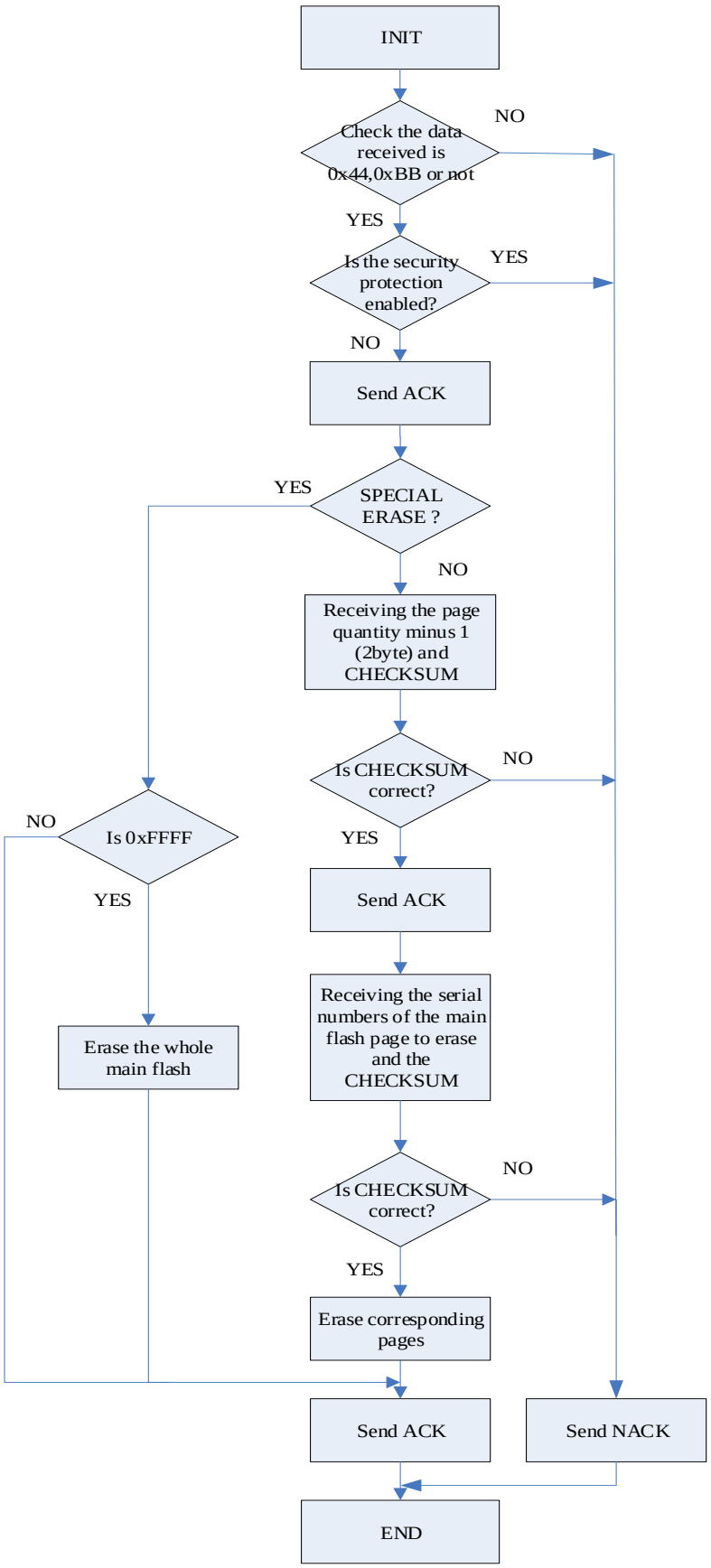
5th BYTE: CHECKSUM of the 3rd and 4th bytes

Wait for ACK

DATA FLOW: The serial numbers of the main flash pages to be erased (2 bytes on each number) and the CHECKSUM

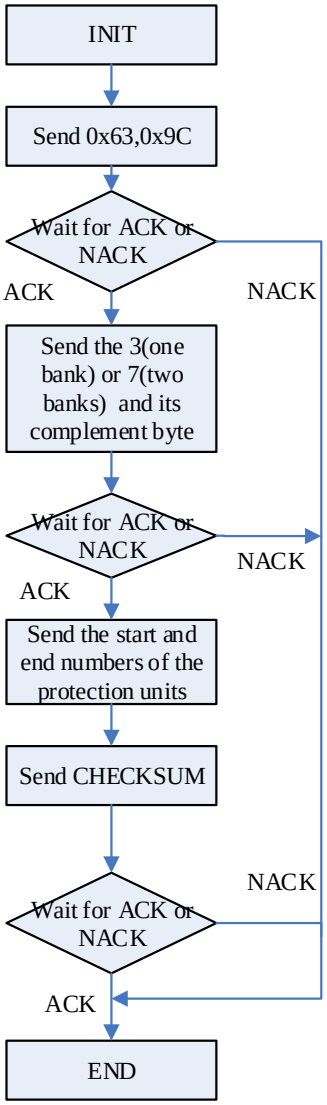
Wait for ACK

Figure 8-2. ERASE command subroutine flow (bootloader)



9 ENABLE ERASE/PROGRAM PROTECTION COMMAND

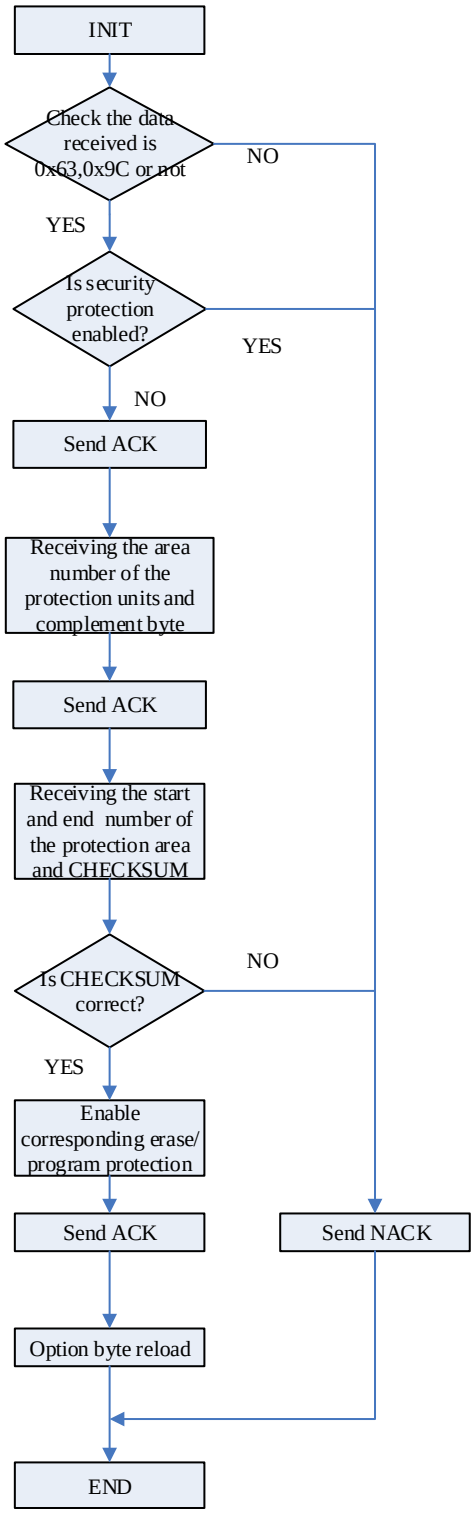
Figure 9-1. ENABLE ERASE/PROGRAM PROTECTION command subroutine flow (PC)



Notes:

- If there is only bank0 in flash, PC send 3, and then send start and end number of two areas successively. While If there are two banks in flash, PC send 7, send start and end number of two areas successively for bank0, and then send start and end number of two areas successively for bank1.
- 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 9-2. ENABLE ERASE/PROGRAM PROTECTION command subroutine flow (bootloader)



10 DISABLE ERASE/PROGRAM PROTECTION COMMAND

Figure 10-1. DISABLE ERASE/PROGRAM PROTECTION command subroutine flow (PC)

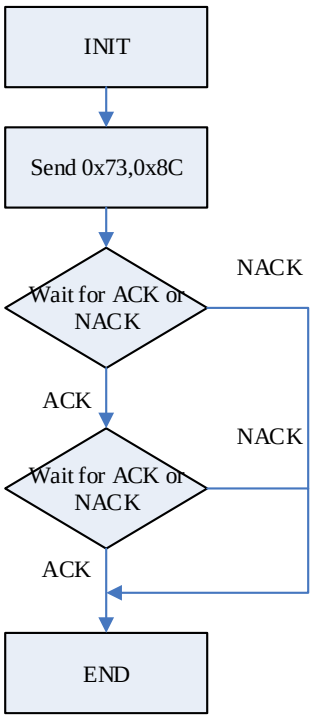
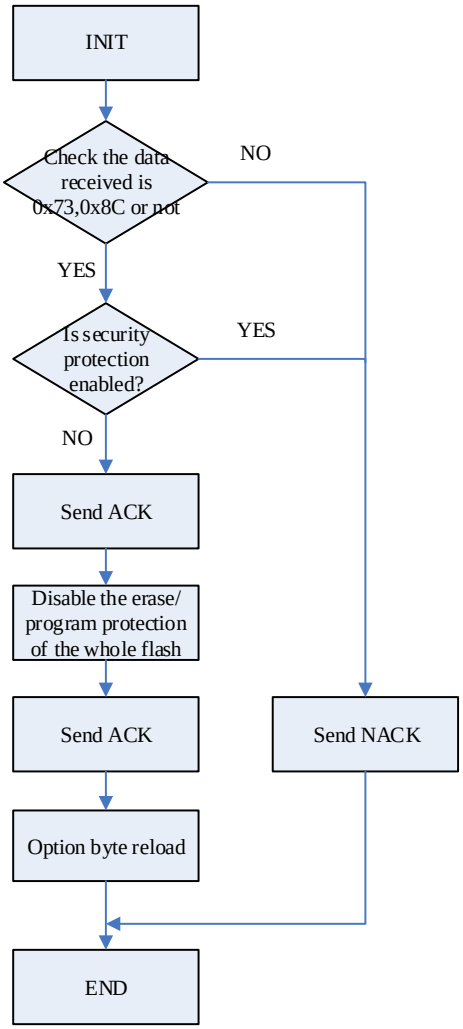


Figure 10-2. DISABLE ERASE/PROGRAM PROTECTION command subroutine flow (bootloader)



11 ENABLE SECURITY PROTECTION COMMAND

Figure 11-1. ENABLE SECURITY PROTECTION command subroutine flow (PC)

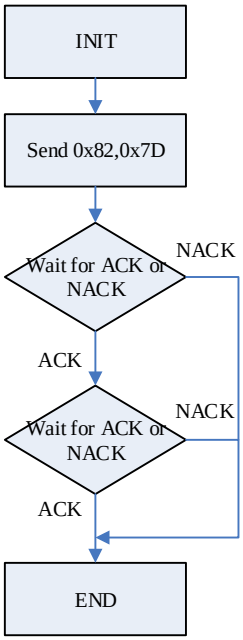
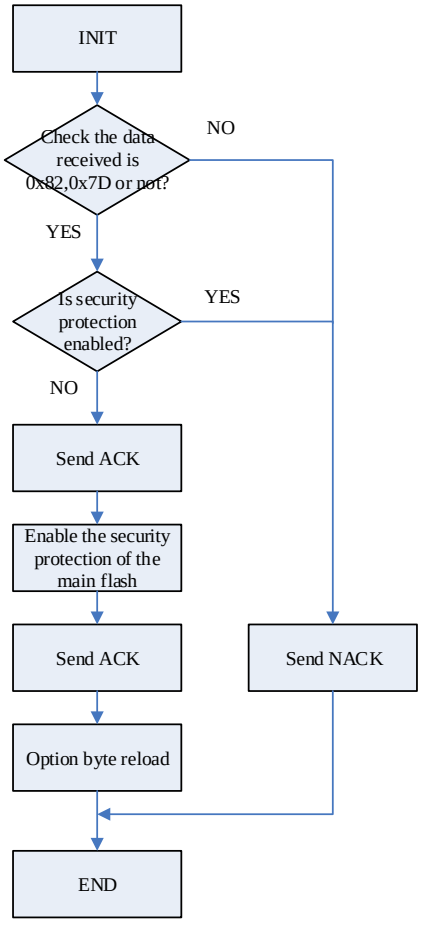


Figure 11-2. ENABLE SECURITY PROTECTION command subroutine flow (bootloader)



12 DISABLE SECURITY PROTECTION COMMAND

Figure 12-1. DISABLE SECURITY PROTECTION command subroutine flow (PC)

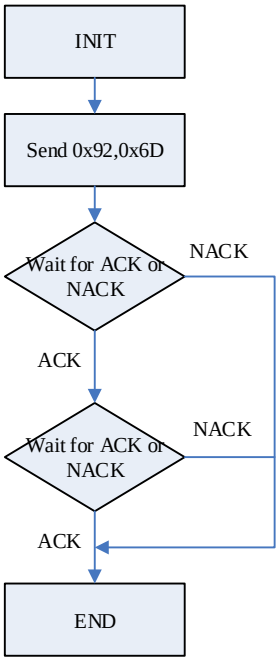
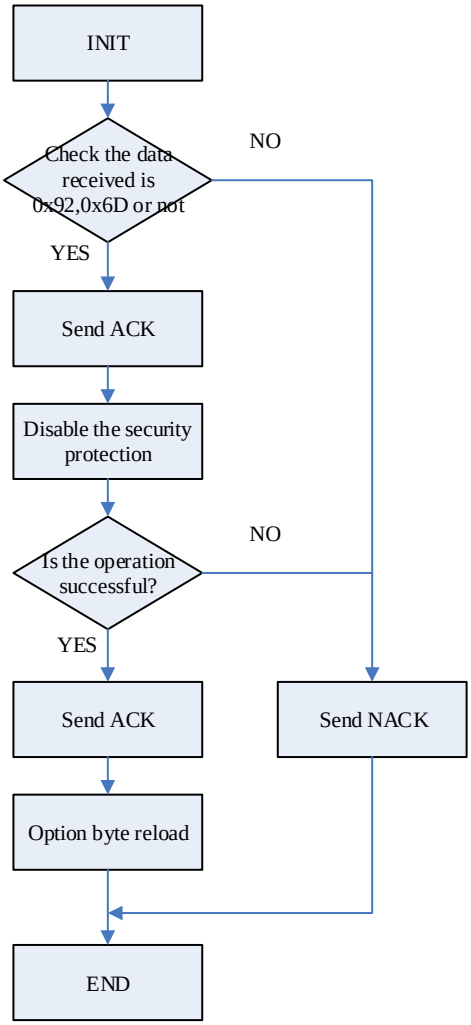


Figure 12-2. DISABLE SECURITY PROTECTION command subroutine flow (bootloader)



13 Revision history

Table 13-1. Revision history

Revision No.	Description	Date
1.0	Initial Release, based on GD32G5xx series	Oct 16, 2024

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.

© 2024 GigaDevice – All rights reserved