User guide

# UG_N32G030 BOOT user guide V1.1

**Notice**

This is a document that is not convenient for National Technology to disclose, and it contains some confidential information. Please return it to National Technology before signing any confidentiality agreement or at the unilateral request of National Technology. Any non-national technical client shall not use or refer to this document.

If you got this document, please note:

- Do not disclose the content of the document
- Do not reproduce all or part of the content of the document
- Do not modify all or part of the document content

This document must be destroyed in the following cases

- National Technology has provided an updated version
- The non-disclosure agreement has not been signed or the non-disclosure agreement has expired
- The appointee resigns

***To our clients:***

We are constantly improving the quality of our products and documentation. While every effort has been made to ensure that the descriptions in this document are accurate, there may be errors that we have not been aware of. If you find any questions or omissions in the document, please contact us in time. Your understanding and support will make this document more complete.

# CONTENTS

# 1. Overview

## BOOT briefly

The firmware program of the MCU, namely BOOT, mainly provides user program download, API and other functions.

This document describes the functions, implementation, and usage of the N32G030 series MCU BOOT in detail. The N32G030 series MCU have a maximum of 64KB of FLASH memory, 3KB of BOOT memory, and 8KB of SRAM.

## 1.1. BOOT function definition

◆ **User program download function**

- Support USART (USART1, using GPIO for PA9-TX, PA10-RX, the default initial baud rate is 9600, support baud rate instruction setting, Supported baud rates 4800, 9600, 14400, 19200, 38400, 57600, 115200, 128000, 256000, 576000, 923076);
- Support download data CRC32 verification;
- Support software reset MCU;
- Support jump to user program.

# 2. BOOT process and command processing

N32G030 series MCU firmware program BOOT, support to download user programs and data through the USART interface. The following describes the command processing flow diagram.

## 2.1. Start the process

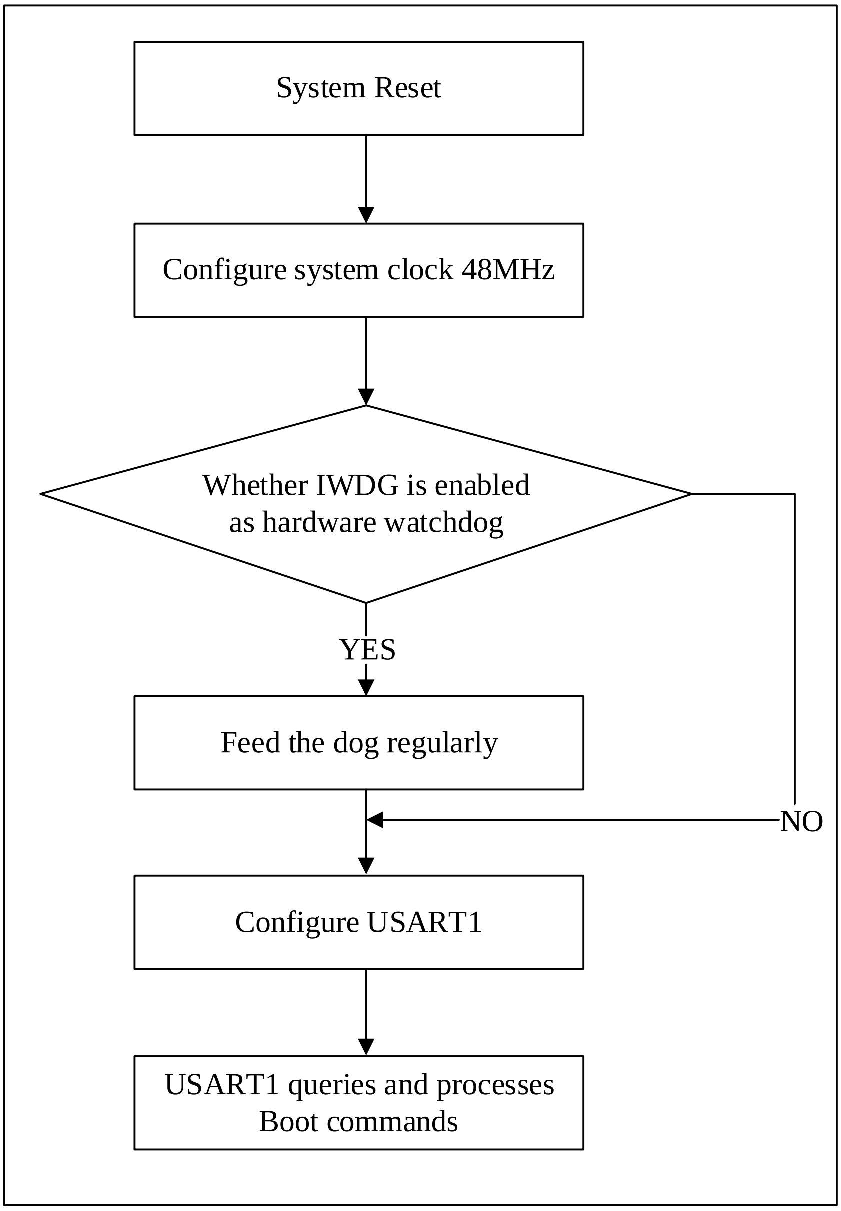
System Reset
Configure system clock 48MHz
Whether IWDG is enabled as hardware watchdog
YES
NO
Feed the dog regularly
Configure USART1
USART1 queries and processes Boot commands

Figure 2.1 Boot Startup flow

Serial port startup process:

1) The default initial baud rate is 9600bps. The host computer uses the command CMD_SET_BR to send the baud rate set by the user to the MCU at 9600bps according to the baud rate set by the user (for example, 115200bps);.

2) After receiving the CMD_SET_BR, the MCU responds and sets the baud rate of the serial port to the value set by the user (for example, 115200bps).

3) The host computer sets the baud rate of the serial port to the value set by the user (for example, 115200bps), and then communicates normally.

## 2.2. Commands and data structure

### 2.2.1. The list of commands

Table 2.1 Command definition

| Name of the command | Value | Instruction |
|---|---|---|
| CMD_SET_BR | 0x01 | Set the baud rate of the serial port (Valid only when serial ports are used) |
| CMD_GET_INF | 0x10 | Read MCU model index, BOOT version number, MCU ID |
| CMD_FLASH_ERASE | 0x30 | Erase FLASH |
| CMD_FLASH_DWNLD | 0x31 | Download user programs to FLASH |
| CMD_DATA_CRC_CHECK | 0x32 | CRC verification download user program |
| CMD_OPT_RW | 0x40 | Read/configure option bytes (including read protection level, FLASH page write protection, DATA0/1 configuration,) |
| CMD_SYS_RESET | 0x50 | The system reset |
| CMD_APP_GO | 0x51 | Jump to user area to execute the program |

### 2.2.2. Data structure

This section describes some conventions described in the following sections. "<>" represents fields that must be included, and "()" represents the fields that included according to different parameters.

**Upper and lower instruction data structure**

1. Upper instruction structure:

<CMD_H + CMD_L + LEN + Par> + (DAT).

CMD_H indicates the level-1 command field, and CMD_L indicates the level-2 command field. LEN indicates the length of data to be sent.Par represents a four-byte command parameter; DAT represents the specific data sent from the upper level instruction to the lower level;

2. Lower response structure:

< CMD_H + CMD_L + LEN > + (DAT) + <CR1+CR2>.

CMD_H indicates the level-1 command field, and CMD_L indicates the level-2 command field. The command fields at the lower level are the same as those at the upper level. LEN indicates the length of data to be sent. DAT indicates the specific data that the lower layer replies to the upper layer. CR1+CR2 indicates the command execution result returned to the upper layer. If the level-1 and level-2 command fields do not belong to any command, BOOT replies CR1=0xBB and CR2 = 0xCC.

**Command data structures supported by the serial port:**

1. The host computer issues the upper instruction:

STA1 + STA2 + {Upper instruction structure} + XOR.

STA1 and STA2 are the start bytes of commands sent through the serial port. STA1=0xAA and STA2=0x55. Used for MCU identification upper computer to send serial data stream.

XOR represents the XOR operation value of the previous command byte (STA1 + STA2 + {Upper instruction structure}).

2. The upper computer receives the lower response:

STA1 + STA2 + {Lower response structure} + XOR.

STA1 and STA2 are the start bytes of commands sent through the serial port. STA1=0xAA and STA2=0x55. It is used for the host computer to identify MCU and send serial port data stream

XOR represents the XOR operation value of the previous command byte (STA1 + STA2 + {Lower response structure}).

*Note: In BOOT V1.0 (for the command of obtaining version information, see section 2.3.2), XOR does not operate on CR2 but only on bytes preceding CR2, that is (STA1 + STA2 + {< CMD_H + CMD_L + LEN > + (DAT) + <CR1 >})*

## 2.3. Command description

### 2.3.1. CMD_ SET_BR

This command is used to change the baud rate of the serial port.

**Upper instruction:**

| bit<br>byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x01 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data: 0x00,0x00 | | | | | | | |
| 4~7(Par) | Par[0~3] : Set baud rate parameters | | | | | | | |
| (DAT) | None | | | | | | | |

- Par[0~3], the serial port baud rate can be set to a typical value;

| Par[0~3] | Toggle specified baud rate (BPS) |
|---|---|
| 0x000E15C4 | 923076 |
| 0x0008CA00 | 576000 |
| 0x0003E800 | 256000 |
| 0x0001F400 | 128000 |
| 0x0001C200 | 115200 |
| 0x0000E100 | 57600 |
| 0x00009600 | 38400 |
| 0x00004B00 | 19200 |
| 0x00003840 | 14400 |
| 0x00002580 | 9600 |
| 0x000012C0 | 4800 |

- Reserved value: 0x00;

**Lower response:**

| bit / byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x01 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data: 0x00,0x00 | | | | | | | |
| (DAT) | None | | | | | | | |
| 4(CR1) | Status byte 1 | | | | | | | |
| 5(CR2) | Status byte 2 | | | | | | | |

- Status bytes (CR1 and CR2) are divided into the following types according to command execution:

1. Return success: status flag bit (0xA0, 0x00).
2. Return failure: status flag bit (0xB0, 0x00).

### 2.3.2. CMD_GET_INF

The function provided by this command is to read the BOOT version number, MCU model index, and MCU ID.

**Upper instructions:**

| bit / byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x10 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3 (LEN) | Length of send data | | | | | | | |
| 4~7(Par) | Reserved | | | | | | | |
| (DAT) | None | | | | | | | |

- Reserved value: 0x00.
- LEN Send data length: 0x00(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] +(LEN[1]<<8).

**Lower response:**

| byte \ bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x10 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3 (LEN) | The length of the data | | | | | | | |
| 4~54(DAT) | BOOT version, MCU model index, MCU ID | | | | | | | |
| 55(CR1) | Status byte 1 | | | | | | | |
| 56(CR2) | Status byte 2 | | | | | | | |

- The procedure byte (CMD_H) corresponds to the upper instruction (CMD_H).
- LEN is the data length: 0x33(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8).
- DAT[0] : 0x01, reserved bit
- DAT[1] : 0xXY, BOOT version (BCD code)
- DAT[2] : BOOT command version
- DAT[3~50] : 48Byte
    1. DAT[3~18] : 16Byte UCID (for details about the UCID, see the user manual).
    2. DAT[19-30] : 12Byte MCU ID(UID) (for details, see the user manual).
    3. DAT[31~34]: 4Byte DBGMCU_IDCODE (for details about DBGMCU_IDCODE, see the user manual).
    4. DAT[35~50] : 16 bytes other information
- Status bytes (CR1 and CR2) are divided into the following types according to command execution:
    1. Return success: status flag bit (0xA0, 0x00).
    2. Return failure: status flag bit (0xB0, 0x00).

### 2.3.3. CMD_FLASH_ERASE

BOOT provides the function of erasing FLASH in units of pages. The erasure page address number and page number are provided by the user. The erased FLASH space cannot exceed the entire FLASH space, and at least 1 page (512Byte) is erased.

**Upper instruction:**

| byte \ bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x30 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data (0) | | | | | | | |
| 4~7(Par) | Page address number 2 bytes: 0 to 255<br>Page Number 2 bytes :1 to 256 | | | | | | | |
| (DAT) | None | | | | | | | |

- CMD_L: erases the partition number

  0x00

- LEN Send data length: 0x10(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8).

- The erase address and range consist of four bytes in the Par field

  Par[0~1] : page address number 2 bytes (0~255)

  Page address number = Par [0] + Par [1]<<8;

  Par[2~3] : Page number 2 bytes (1~256)

  Page number = Par [2] + Par [3]<<8;

  The first address of page 0 is 0x0800_0000. The number of subsequent pages is increased by 1, and the first address is added to 0x200.

  For example:

  The first address of page 1 is 0x0800_0000 + 1*0x200 = 0x0800_0200

  The first address of page 2 is 0x0800_0000 + 2*0x200 = 0x0800_0400

  The entire address range erased

  For example, the page address is 0x01 and the number of pages is 0x02

  Erasing address range:

  (0x0800_0000 + 1*0x200) ~ (0x0800_0000 + 1*0x200 + 2*0x200)

  That is, (first address of the page number) to (first address of the page number + (number of pages)*(page size))

**Lower response:**

| bit / byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x30 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data | | | | | | | |
| (DAT) | None | | | | | | | |
| 4(CR1) | Status byte 1 | | | | | | | |
| 5(CR2) | Status byte 2 | | | | | | | |

- LEN Send data length: 0x00(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8).
- Status bytes (CR1 and CR2) are divided into the following types according to command execution:

1. Return success: status flag bit (0xA0, 0x00).
2. Return failure: status flag bit (CR1, CR2).

(1), (0xB0, 0x00) : return failure;

(2), (0xB0, 0x30) : The erased FLASH page is protected by RDP;

(3), (0xB0, 0x31) : The erased FLASH page is protected by WRP;

(4), (0xB0, 0x32) : Erase FLASH page is protected by partition;

(5), (0xB0, 0x33) : Erase FLASH page range across partitions;

(6), (0xB0, 0x34) : The FLASH address range is out of bounds (that is, it exceeds the size of the entire FLASH);

(7), (0xB0, 0x35) : Download FLASH start address is not 16 bytes aligned;

(8), (0xB0, 0x36) : The downloaded FLASH data length is not a multiple of 16;

(9), (0xB0, 0x37) : Failed to erase the FLASH.

### 2.3.4. CMD_FLASH_DWNLD

This command allows the user to download code into the specified FLASH. Data length must be 16 bytes aligned (0x00 automatically added by the host computer when the length is less than 16 bytes.), all provided by upper-layer commands. Plaintext download.

**Upper instruction:**

| bit<br>byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x31 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data | | | | | | | |
| 4~7(Par) | Start address for downloading the FLASH | | | | | | | |
| 8~23+N(DAT) | DAT[16~16+N] : Specific data downloaded<br>DAT[16+N+1~16+N+4] : 4 byte CRC32 check value of data | | | | | | | |

- CMD_L: erases the partition number

  0x00.

- LEN Send data lenth: 0xXX(LEN[0]), 0xXX(LEN[1]), LEN = LEN[0] + (LEN[1]<<8)
- Par [0 ~ 3]: download the starting Address of the FLASH, synthetic rules to Address = Par[0] | Par[1]<<8 | Par[2]<<16 | Par[3]<<24.
- DAT[0~15]: Reserved, all is 0
- DAT[16~16+N]: Specific data to be downloaded, total number is N+1

  USART: contains a maximum of 128 bytes. 15<=N<=143. N+1 must be a multiple of 16.

  DAT[16+N+1~16+N+4]: 4 byte CRC32 check value of data

**Lower response:**

| bit<br>byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x31 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2(LEN) | Length of send data | | | | | | | |
| (DAT) | None | | | | | | | |
| 3(CR1) | Status byte 1 | | | | | | | |
| 4(CR2) | Status byte 2 | | | | | | | |
| 5(XOR) | XOR result | | | | | | | |

- LEN Send data length: 0x00(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8).
- Status bytes (CR1 and CR2) are divided into the following types according to command execution:

1. Download success: status flag bit (0xA0, 0x00).
2. Download failed: status flag bit (CR1, CR2).

(1), (0xB0, 0x00): Return failure;

(2), (0xB0, 0x30): The downloaded FLASH address is protected by RDP;

(3), (0xB0, 0x31): The downloaded FLASH address is protected by WRP;

(4), (0xB0, 0x32): The downloaded FLASH address is protected by partition

(5), (0xB0, 0x33): Download FLASH address range across partitions;

(6), (0xB0, 0x34): Download FLASH address range is out of bounds (refers to beyond the size of the entire FLASH);

(7), (0xB0, 0x35): Download FLASH start address is not 16 bytes aligned;

(8), (0xB0, 0x36): The downloaded FLASH data length is not a multiple of 16;

(9), (0xB0, 0x37): Failed to program the FLASH.

### 2.3.5. CMD_DATA_CRC_CHECK

This command is used to check whether the downloaded data is correct. Considering the download speed and low probability of download failure, the CRC check is performed after the downloaded data is complete. The upper command must provide the CRC value, start address, and check length of the downloaded data.

**Upper instructions:**

| bit / byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x32 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data | | | | | | | |

| | |
|---|---|
| 4~7(Par) | 32-bit CRC check value |
| 8~31(DAT) | DAT [0~15]: Reserved<br>DAT[16~19]: Start address of the verification<br>DAT[20~23]: Verification length (in bytes, minimum length 512B) |

- CMD_L: indicates the verification partition number

  0 x00.

- LEN Send data length: 0x18(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8).

- Par [0 ~ 3]: 32 bit CRC checksum value, the synthetic rules for CRC32 = CRC32 = Par[0] | Par[1]<<8 | Par[2]<<16 | Par[3]<<24.

- CMD_L = 0x00:16 bytes are all 0x00.

- DAT [0~15]: reserved, all is 0

- DAT [16 ~ 19]: check the starting Address, the synthesis rules to Address = DAT [16] | DAT [17] < < 8 | DAT [18] < < 16 | DAT [19] < < 24, the Address can only be in the range of the FLASH.

- DAT [20 ~ 23]: check length, its synthesis rules for CRC_LEN = DAT [20] | DAT [21] < < 8 | DAT [22] < < 16 | DAT [23] < < 24, CRC_LEN is only within the effective range, length is larger than 2 KB, and is a multiple of 16.

**Lower response:**

| bit<br>byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x32 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data | | | | | | | |
| (DAT) | None | | | | | | | |
| 4(CR1) | Status byte 1 | | | | | | | |
| 5(CR2) | Status byte 2 | | | | | | | |

- LEN send data length: 0x00(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8).

- Status bytes (CR1 and CR2) are divided into the following types according to command execution:

1. Verification succeeded: status flag bit (0xA0, 0x00).
2. Check failure: status flag bit (CR1, CR2)

(1), (0xB0, 0x00): return failure;

(2), (0xB0, 0x32): CRC check addresses are protected by partitions.

(3), (0xB0, 0x33): CRC check address range is across partitions;

(4), (0xB0, 0x34): CRC check address range is out of bounds (refers to the size of the entire FLASH);

(5), (0xB0, 0x35): CRC check address is not 16-byte alignment;

(6), (0xB0, 0x36): The CRC check length is not a multiple of 16 or less than 512 BYTES.

(7), (0xB0, 0x38): CRC verification failed.

### 2.3.6. CMD_OPT_RW

This command is used for option byte read and write (including read protection level, FLASH page write protection, datA0/1 configuration, and USER configuration).

**Upper instruction:**

| byte \ bit | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x40 Level-1 command field | | | | | | | |
| 1(CMD_L) | Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data | | | | | | | |
| 4~7(Par) | | | | | | | | |
| 8~23(DAT) | Option byte configures 16 bytes | | | | | | | |

- CMD_L Level-2 command field:
  1. 0x00: Gets option bytes.
  2. 0x01: Configuration option byte.
  3. 0x02: Configuration option byte, reset again.
- LEN Send data length: 0x14(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8).

- DAT[0~15]: Option bytes 16 bytes

RDP, nRDP, USER, nUSER, Data0, nData0, Data1, nData1, WRP0, nWRP0, WRP1, nWRP1, RDP2, nRDP2, Reserved, nReserved;

1. CMD_L = 0x00: all values are 0x00.
2. CMD_L = 0x01/0x02: Configuration option bytes are the values to be written.

**Lower response:**

| bit / byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x40 Level-1 command field | | | | | | | |
| 1(CMD_L) | Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data | | | | | | | |
| 4~19(DAT) | Option byte configures 16 bytes | | | | | | | |
| 24(CR1) | Status byte 1 | | | | | | | |
| 25(CR2) | Status byte 2 | | | | | | | |

- LEN send data length: 0x14(LEN[0]), 0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8).

- DAT[0~15]: The current option contains 16 bytes

RDP, nRDP, USER, nUSER, Data0, nData0, Data1, nData1, WRP0, nWRP0, WRP1, nWRP1, RDP2, nRDP2, Reserved, nReserved;

- Status bytes (CR1 and CR2) are divided into the following types according to command execution:

1. Return success: status flag bit (0xA0, 0x00).
2. Check failure: status flag bit (CR1, CR2)

(1), (0xB0, 0x00): return failure;

(2), (0xB0, 0x39): The matched area is sealed and the read protection level is not allowed to be reduced from L1 to L0;

### 2.3.7. CMD_SYS_RESET

This command is used to reset the BOOT program.

**Upper instruction:**

| bit<br>byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x50 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data | | | | | | | |
| 4~7(Par) | Reserved | | | | | | | |
| (DAT) | None | | | | | | | |

- Reserved value: 0x00;

**Lower response:**

| bit<br>byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x50 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3(LEN) | Length of send data | | | | | | | |
| (DAT) | None | | | | | | | |
| 4(CR1) | Status byte 1 | | | | | | | |
| 5(CR2) | Status byte 2 | | | | | | | |

- Status bytes (CR1 and CR2) are divided into the following types according to command execution:

1. Return success: status flag bit (0xA0, 0x00).
2. Return failure: status flag bit (0xB0, 0x00).

### 2.3.8. CMD_APP_GO

This command is used to reset the program entry address (0x0800_0000) after the BOOT program is downloaded to the FLASH.

**Upper instruction:**

| bit<br>byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x51 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |

| | |
|---|---|
| 2~3(LEN) | Length of send data |
| 4~7(Par) | Reserved |
| (DAT) | None |

- Reserved value: 0x00;

**Lower response:**

| bit<br>byte | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| 0(CMD_H) | 0x51 Level-1 command field | | | | | | | |
| 1(CMD_L) | 0x00 Level-2 command field | | | | | | | |
| 2~3 (LEN) | Length of send data | | | | | | | |
| (DAT) | None | | | | | | | |
| 4(CR1) | Status byte 1 | | | | | | | |
| 5(CR2) | Status byte 2 | | | | | | | |

- Status bytes (CR1 and CR2) are divided into the following types according to command execution:

1. Return success: status flag bit (0xA0, 0x00).
2. Return failure: status flag bit (0xB0, 0x00).

## 2.4. Return the status word description

### 2.4.1. Returns the success status word

Return success: status flag bit (0xA0, 0x00). The command from the upper layer is successfully executed, and the success status is displayed.

Contains the success return value of the read, update, configure, and other commands.

### 2.4.2. Returns the failure status word

Return failure: status flag bits (0xB0, 0x00). Indicates that the command delivered by the upper layer fails to be executed due to other reasons (such as incorrect command acceptance format or timeout). Failure status is returned.

### 2.4.3. Return other status words

The following return status words also return failure. The second byte status word indicates a different error type.

(1), (0xB0, 0x30): Erase/download FLASH page protected by RDP;

(2), (0xB0, 0x31): Erased/downloaded FLASH page is protected by WRP;

(3), (0xB0, 0x32): Erase/download /CRC check address is protected by partition;

(4), (0xB0, 0x33): Erase/download /CRC check address range across partitions;

(5), (0xB0, 0x34): The address range of erase/download /CRC is out of bounds (refers to the size of the entire FLASH);

(6), (0xB0, 0x35): Erase/download /CRC start address is not 16-byte alignment;

(7), (0xB0, 0x36): The length of the download /CRC check data is not a multiple of 16;Data length indicates the length of erasing FLASH, or the length of downloading code to FLASH, or the length of checking FLASH CRC values;

(8), (0xB0, 0x37): Failed to erase/download FLASH programming;

(9), (0xB0, 0x38): CRC verification fails;

(10), (0xB0, 0x39): Partition sealing has been configured, and read protection level is not allowed to be reduced from L1 to L0;

(11), (0xBB, 0xCC): The level-1 and level-2 command fields do not belong to any command.

# 3. Version history

| Version | Date | Note |
|---|---|---|
| V1.0 | 2021-09-24 | Create a document |
| V1.1 | 2022-06-17 | 1. Delete CMD_USER_SEAL boot command；<br>2. Modify notice description, header and footer；<br>3. Modify the error of URART in the second chapter brief description, and correct it to USART<br>4. Section 2.2.2. Add the note of XOR operation |
| | | |

# 4. NOTICE

This document is the exclusive property of Nations Technologies Inc. (Hereinafter referred to as NATIONS). This document, and the product of NATIONS described herein (Hereinafter referred to as the Product) are owned by NATIONS under the laws and treaties of the People’s Republic of China and other applicable jurisdictions worldwide.

NATIONS does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. Names and brands of third party may be mentioned or referred thereto (if any) for identification purposes only.

NATIONS reserves the right to make changes, corrections, enhancements, modifications, and improvements to this document at any time without notice. Please contact NATIONS and obtain the latest version of this document before placing orders.

Although NATIONS has attempted to provide accurate and reliable information, NATIONS assumes no responsibility for the accuracy and reliability of this document.

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. In no event shall NATIONS be liable for any direct, indirect, incidental, special, exemplary, or consequential damages arising in any way out of the use of this document or the Product.

NATIONS Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, “Insecure Usage”.

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at user’s risk. User shall indemnify NATIONS and hold NATIONS harmless from and against all claims, costs, damages, and other liabilities, arising from or related to any customer’s Insecure Usage.

Any express or implied warranty with regard to this document or the Product, including, but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement are disclaimed to the fullest extent permitted by law.

Unless otherwise explicitly permitted by NATIONS, anyone may not use, duplicate, modify, transcribe or otherwise distribute this document for any purposes, in whole or in part.