

使用指南

UG_N32G05x 系列 MCU BOOT 接口指令使用指南

简介

使用指南主要描述 N32G05x 系列 MCU 的 BOOT 接口指令，便于使用国民技术 BOOT Loader 进行下载开发。

目录

1. BOOT 简述	1
1.1 BOOT 功能定义.....	2
2. BOOT 流程及命令处理	3
2.1. 命令及数据结构.....	3
2.1.1. 命令列表.....	3
2.1.2. 数据结构.....	3
2.1.3. 命令示例.....	4
2.2. 命令说明.....	5
2.2.1. CMD_SET_BR.....	5
2.2.2. CMD_GET_INF.....	6
2.2.3. CMD_FLASH_ERASE.....	8
2.2.4. CMD_FLASH_DWNLD.....	10
2.2.5. CMD_DATA_CRC_CHECK.....	11
2.2.6. CMD_OPT_RW.....	14
2.2.7. CMD_USERX_OP.....	15
2.2.8. CMD_SYS_RESET.....	19
2.2.9. CMD_APP_GO.....	20
2.3. 返回状态字说明.....	21
2.3.1. 返回成功状态字.....	21
2.3.2. 返回失败状态字.....	21
2.3.3. 返回其他状态字.....	21
3. BOOT 使用说明	23
3.1. 上位机控制流程.....	23
4. 历史版本	24
5. 声明	25

1. BOOT简述

芯片的固件程序即 BOOT 主要提供用户程序下载，API 等功能。

本文档详细描述了 N32G05X 系列芯片 BOOT 的功能、实现及使用介绍。N32G05X 系列芯片的 Main FLASH 存储区最大为 128KB，Data FLASH 存储区最大为 8KB。

1.1 BOOT功能定义

◆ 用户程序下载功能

- 支持 UART (UART1, 默认 PA9/PA10, 可通过选项字节 USER6[1:0]配置为 PB10/PB11,PD10/PD11,PA2/PA3, 波特率协商);
- 支持下载数据 CRC32 校验;
- 支持明文下载;
- 支持 FLASH 分区;
- 支持上电 BOOT 自校验;
- 支持跳转到 Main flash/SRAM 用户区执行;
- 支持软件复位芯片。

2. BOOT流程及命令处理

N32G05X 系列芯片的固件程序 BOOT，支持通过 UART 接口下载用户程序和数据。下面阐述相关命令处理流程。

2.1. 命令及数据结构

2.1.1. 命令列表

Table2-1 命令定义

命令名称	键值	说明
CMD_SET_BR	0x01	设置串口波特率（仅使用串口时有效）
CMD_GET_INF	0x10	读取芯片型号索引、BOOT 版本号、芯片 ID
CMD_FLASH_ERASE	0x30	擦除 FLASH
CMD_FLASH_DWNLD	0x31	下载用户程序到 FLASH
CMD_DATA_CRC_CHECK	0x32	CRC 校验下载用户程序
CMD_OPT_RW	0x40	读取/配置选项字节（包含了读保护等级、FLASH 页写保护、Data0/1 配置、USER 配置）
CMD_USERX_OP	0x41	获取分区 USERX 大小，配置分区 USERX 大小
CMD_SYS_RESET	0x50	系统复位
CMD_APP_GO	0x51	跳转到用户区执行程序

2.1.2. 数据结构

这里介绍下文阐述中的一些约定，其中，“<>”代表必须包含的字段，“()”代表根据参数不同包含的字段。

上下层指令数据结构

1、上层指令结构：

<CMD_H + CMD_L + LEN + Par> + (DAT)。

CMD_H 代表一级命令字段，CMD_L 代表二级命令字段；LEN 代表发送数据长度；Par 代表 4 个字节命令参数；DAT 代表上层指令往下层发送的具体数据；

2、下层应答结构：

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

CMD_H 代表一级命令字段，CMD_L 代表二级命令字段，下层的命令字段和对应上层

的命令字段相同；LEN 代表发送数据长度；DAT 代表下层向上层应答的具体数据；CR1+CR2 代表向上层返回的指令执行结果，若上层发送命令一级、二级命令字段不属于任何命令，BOOT 回复 CR1=0xBB，CR2 = 0xCC。

串口支持的命令数据结构：

1、上位机下发上层指令：

STA1 + STA2 + {上层指令结构} + XOR。

STA1 和 STA2 是串口发送命令的起始字节，STA1=0xAA，STA2=0x55。用于芯片识别上位机发送串口数据流。

XOR 代表之前命令字节的异或运算值（STA1 + STA2 + {上层指令结构}）。

2、上位机接收下层应答：

STA1 + STA2 + {下层应答结构} + XOR。

STA1 和 STA2 是串口发送命令的起始字节，STA1=0xAA，STA2=0x55。用于上位机识别芯片发送串口数据流

XOR 代表之前命令字节的异或运算值（STA1 + STA2 + {下层应答结构}）。

2.1.3. 命令示例

下表将每个命令类型都选择了一个命令作为示例供参考，命令详细说明请参见命令说明章节。

命令名称	命令示例	命令结构 ⁽²⁾							
		STA1	STA2	CMD _H	CMD _L	LEN[1: 0]	Par[3: 0]	DAT	XOR
CMD_SET_BR	设置串口波特率 4800	AA	55	01	00	00,00	00,00, 12,C0	空 ⁽¹⁾	2C
CMD_GET_INFO	读取芯片信息	AA	55	10	00	00,00	00,00, 00,00	空 ⁽¹⁾	EF
CMD_FLASH_ERASE	擦除 DATA FLASH 第 0 页	AA	55	30	03	00,00	00,00, 01,00	空 ⁽¹⁾	CD

CMD_FLAS H_DWNLD	下载 DATA FLASH 第 0 页 16 个字 节	AA	55	31	03	24,00	00,10, FF,1F	32 个 0x00, C8,22, 2D,55	8B
CMD_DATA _CRC_CHEC K	CRC 校验 DATA FLASH 第 0 页	AA	55	32	03	18,00	实际 CRC 值 4 个字 节	16 个 0x00, 00,10, FF,1F , 00,02, 00,00	根据 CRC 值 得异或 值
CMD_OPT_R W	获取选项字 节	AA	55	40	00	0E,00	00,00, 00,00	14 个 0x00	B1
CMD_USER X_OP	读 取 USER1 配 置	AA	55	41	00	00,00	00,00, 00,00	空 ⁽¹⁾	BE
CMD_SYS_R ESET	软件复位 boot 程序	AA	55	50	00	00,00	00,00, 00,00	空 ⁽¹⁾	AF
CMD_APP_G O	跳 转 到 Main FLASH	AA	55	51	00	00,00	00,00, 00,00	空 ⁽¹⁾	AE

注:

1. 空表示没有数据
2. 上层指令详解见命令说明章节

2.2. 命令说明

2.2.1. CMD_SET_BR

该命令用于修改串口波特率。

上层指令:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x01 一级命令字段							
1(CMD_L)	0x00 二级命令字段							
2~3(LEN)	发送数据长度: 0x00,0x00							
4~7(Par)	Par[0~3]: 设置波特率参数							
(DAT)	无							

●Par[0~3], 串口波特率协商设置值可以设定最大, 设定范围为 2.4Kbps~923.076Kbps, 默认波特率为 9600bps; $\text{baudrate} = (\text{Par}[0] \ll 24) + (\text{Par}[1] \ll 16) + (\text{Par}[2] \ll 8) + \text{Par}[3]$ 。

●保留值: 0x00;

底层应答:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x01 一级命令字段							
1(CMD_L)	0x00 二级命令字段							
2~3(LEN)	发送数据长度: 0x00,0x00							
(DAT)	无							
4(CR1)	状态字节 1							
5(CR2)	状态字节 2							

●状态字节(CR1、CR2)根据命令执行情况分为:

- 1.返回成功: 状态标志位(0xA0、0x00)。
- 2.返回失败: 状态标志位(0xB0、0x00)。

下面是波特率协商支持的波特率值(√ 表示支持, / 表示不支持):

时钟参数 (MHz)	波特率											
	2400	4800	9600	14400	19200	38400	57600	115200	128000	256000	576000	923076
HSI	√	√	√	√	√	√	√	√	√	√	√	√

2.2.2. CMD_GET_INF

该命令提供的功能是读取 BOOT 版本号、芯片型号索引、芯片 ID、芯片系列化信息共 4 种信息。

上层指令:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x10 一级命令字段							
1(CMD_L)	0x00 二级命令字段							
2~3 (LEN)	发送数据长度							
4~7(Par)	保留							
(DAT)	无							

●保留值：0x00。

●LEN 发送数据长度：0x00(LEN[0])、0x00(LEN[1])， $LEN = LEN[0] + (LEN[1] \ll 8)$ 。

底层应答：

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x10 一级命令字段							
1(CMD_L)	0x00 二级命令字段							
2~3 (LEN)	数据长度							
4~54(DAT)	BOOT 版本号、芯片型号索引、芯片 ID							
55(CR1)	状态字节 1							
56(CR2)	状态字节 2							

●过程字节(CMD_H)和上层指令中的(CMD_H)对应。

●LEN 是数据长度：0x33(LEN[0])、0x00(LEN[1])， $LEN = LEN[0] + (LEN[1] \ll 8)$ 。

●DAT[0]：0x0B，芯片型号索引

●DAT[1]：0xXY，BOOT 版本号(BCD 码)

0x10：指示 BOOT 使用的命令集版本，表示使用 V1.0 的命令集版本。

●DAT[2]：BOOT 命令集版本

●DAT[3~50] 48Byte

1. DAT[3~18]：16Byte UCID (UCID 的详细定义见用户手册)；
2. DAT[19~30]：12Byte Chip ID(UID) (UID 的详细定义见用户手册)；
3. DAT[31~34]：4Byte DBGMCU_IDCODE (DBGMCU_IDCODE 的详细定义见用户手册)；
4. DAT[35~50]：16Byte 芯片型号；

●状态字节(CR1、CR2)根据命令执行情况分为：

1. 返回成功：状态标志位(0xA0、0x00)。
2. 返回失败：状态标志位(0xB0、0x00)。

2.2.3. CMD_FLASH_ERASE

BOOT 提供以页为单位擦除 FLASH 的功能，擦除页地址编号和和页数由用户提供，擦除的 FLASH 空间不能超过整个 FLASH 空间，且至少擦除 1 个页(512Byte)。

上层指令：

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x30 一级命令字段							
1(CMD_L)	0xXX 二级命令字段							
2~3(LEN)	发送数据长度(0)							
4~7(Par)	页地址编号 2 字节：0~255 页数 2 字节：1~256							

●CMD_L：擦除分区号

- 0x00 = USER1；（分区封口以后可擦除，可通过 FLASH 封口实现不可擦除，具体见 2.2.7 章节）
- 0x01 = USER2；（分区封口以后可擦除，可通过 FLASH 封口实现不可擦除，具体见 2.2.7 章节）
- 0x02 = USER3；（分区封口以后可擦除，可通过 FLASH 封口实现不可擦除，具体见 2.2.7 章节）
- 0x03 = Data Flash；
- 0x04 = SRAM；支持地址范围 0x20001000~0x2003FFF。SRAM 擦除操作实际不做任何操作，直接返回 OK。
- LEN 发送数据长度：0x10(LEN[0])、0x00(LEN[1])， $LEN = LEN[0] + (LEN[1] \ll 8)$ 。
- 擦除地址和范围由 Par 字段中的 4 个字节构成

Par[0~1]：页地址编号 2 字节(0~255)

页地址编号 = $Par[0] + Par[1] \ll 8$ ；

Par[2~3]：页数 2 字节(1~256)

页数 = $Par[2] + Par[3] \ll 8$ ；

0 号页首地址为 0x0800_0000，以后的页地址编号加 1，首地址累加 0x200。

比如：

1 号页首地址为 $0x0800_0000 + 1 * 0x200 = 0x0800_0200$

2 号页首地址为 $0x0800_0000 + 2 * 0x200 = 0x0800_0400$

整个擦除的地址范围

比如：页地址编号为 0x01，页数为 0x02

则擦除的地址范围：

$(0x0800_0000 + 1 * 0x200) \sim (0x0800_0000 + 1 * 0x200 + 2 * 0x200)$

即（页地址编号的首地址） ~ （页地址编号的首地址 + 页数*页的大小）

底层应答：

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x30 一级命令字段							
1(CMD_L)	二级命令字段：擦除区域							
2~3(LEN)	发送数据长度							
(DAT)	无							
4(CR1)	状态字节 1							
5(CR2)	状态字节 2							

●LEN 发送数据长度：0x00(LEN[0])、0x00(LEN[1])， $LEN = LEN[0] + (LEN[1] \ll 8)$ 。

●状态字节(CR1、CR2)根据命令执行情况分为：

1. 返回成功：状态标志位(0xA0、0x00)。

2. 返回失败：状态标志位(CR1, CR2)。

(1)、(0xB0、0x00)：返回失败；

(2)、(0xB0、0x31)：擦除 FLASH 页被 WRP 保护；

(3)、(0xB0、0x32)：擦除 FLASH 页被分区保护；

(4)、(0xB0、0x33)：擦除 FLASH 页范围跨分区；

(5)、(0xB0、0x34)：擦除 FLASH/SRAM 地址范围越界（指超出整个 FLASH/SRAM

大小）；

(6)、(0xB0、0x37)：擦除 FLASH 失败。

(7)、(0xB0、0x42): FLASH 封口错误, FLASH 已封口, 擦除 FLASH 失败

2.2.4. CMD_FLASH_DWNLD

该命令提供用户下载代码到指定 FLASH 中。数据长度必须 16 字节对齐 (不足上位机自动补 0x00), 都由上层命令提供。明文写入 FLASH。

上层指令:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x31 一级命令字段							
1(CMD_L)	二级命令字段: 下载分区号							
2~3(LEN)	发送数据长度							
4~7(Par)	下载 FLASH 的起始地址							
8~23(DAT)	DAT[0:15]: 保留 (0)							
24~(24+N)(DAT)	DAT[16~16+N]: 下载的具体数据							
(24+N+1)~(24+N+4)(DAT)	DAT[16+N+1~16+N+4]: 数据的 4Byte CRC32 校验值							

●CMD_L: 下载分区号

- 0x00 = USER1; (分区封口以后不可下载, 避免用户下载非法代码读取封口区域代码)
- 0x01 = USER2; (分区封口以后不可下载, 避免用户下载非法代码读取封口区域代码)
- 0x02 = USER3; (分区封口以后不可下载, 避免用户下载非法代码读取封口区域代码)
- 0x03 = Data Flash;
- 0x04 = SRAM; 支持地址范围 0x20001000~0x2003FFF。

●LEN 发送数据长度: 0xXX(LEN[0])、0xXX(LEN[1]), $LEN = LEN[0] + (LEN[1] \ll 8)$

●Par[0~3]: 下载 FLASH 的起始地址, 合成规则为 $Address = Par[0] | Par[1] \ll 8 | Par[2] \ll 16 | Par[3] \ll 24$ 。

●DAT[0~15]: 保留 (0):

●DAT[16~16+N]: 下载的具体数据, 数据总个数为 N+1

UART: 最大 128 个字节, $16 \leq N+1 \leq 128$, N+1 必须为 16 的倍数。

- DAT[16+N+1~16+N+4]: 非加密数据的 4Byte CRC32 校验值

底层应答:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x31 一级命令字段							
1(CMD_L)	二级命令字段: 下载分区号							
2(LEN)	发送数据长度							
(DAT)	无							
3(CR1)	状态字节 1							
4(CR2)	状态字节 2							

- LEN 发送数据长度: 0x00(LEN[0])、0x00(LEN[1]), $LEN = LEN[0] + (LEN[1] \ll 8)$ 。

- 状态字节(CR1、CR2)根据命令执行情况分为:

1. 下载成功: 状态标志位(0xA0、0x00)。
2. 下载失败: 状态标志位(CR1, CR2)。
 - (1)、(0xB0、0x00): 返回失败;
 - (3)、(0xB0、0x31): 下载 FLASH 地址被 WRP 保护;
 - (4)、(0xB0、0x32): 下载 FLASH 地址被分区保护;
 - (5)、(0xB0、0x33): 下载 FLASH 地址范围跨分区;
 - (6)、(0xB0、0x34): 下载 FLASH/SARM 地址范围越界(指超出整个 FLASH/SARM 大小);
 - (7)、(0xB0、0x35): 下载 FLASH 起始地址不是 16 字节对齐;
 - (8)、(0xB0、0x36): 下载 FLASH 数据长度不是 16 的倍数;
 - (9)、(0xB0、0x37): 编程 FLASH 失败;
 - (10)、(0xB0、0x42): FLASH 封口错误, FLASH 已封口, 下载 FLASH 失败。

2.2.5. CMD_DATA_CRC_CHECK

该命令用于校验下载数据是否正确, 考虑到下载速度的因素和下载失败概率比较小, 所以采用数据下载完成后统一进行 CRC 校验, 上层指令需提供下载数据的 CRC 值和校验起始地址以及校验长度。

MMU 封口情况下无法执行 CRC 校验。

上层指令:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x32 一级命令字段							
1(CMD_L)	二级命令字段: 校验分区号							
2~3(LEN)	发送数据长度							
4~7(Par)	32bit CRC 校验值							
8~23(DAT)	DAT[0~15]: 保留 (0)							
24~27(DAT)	DAT[16~19]: 校验起始地址							
28~31(DAT)	DAT[20~23]: 校验长度(单位: 字节, 长度最小 512B)							

●CMD_L: 校验分区号

- 0x00 = USER1, 分区封口以后, 不可进行校验;
- 0x01 = USER2, 分区封口以后, 不可进行校验;
- 0x02 = USER3, 分区封口以后, 不可进行校验;
- 0x03 = Data Flash;
- 0x04 = SRAM; 支持地址范围 0x20001000~0x2003FFF。

●LEN 发送数据长度: $0x18(\text{LEN}[0])$ 、 $0x00(\text{LEN}[1])$, $\text{LEN} = \text{LEN}[0] + (\text{LEN}[1] \ll 8)$;

●Par[0~3]: 32bit CRC 校验值, 其合成规则为 $\text{CRC32} = \text{Par}[0] | \text{Par}[1] \ll 8 | \text{Par}[2] \ll 16 | \text{Par}[3] \ll 24$;

●DAT[0:15]: 保留 (0);

●DAT [16~19]: 校验起始地址, 其合成规则为 $\text{Address} = \text{DAT}[16] | \text{DAT}[17] \ll 8 | \text{DAT}[18] \ll 16 | \text{DAT}[19] \ll 24$, Address 只能是在 FLASH/SRAM 范围内;

●DAT [20~23]: 校验长度, 其合成规则为 $\text{CRC_LEN} = \text{DAT}[20] | \text{DAT}[21] \ll 8 | \text{DAT}[22] \ll 16 | \text{DAT}[23] \ll 24$, CRC_LEN 只能是在有效范围内, 长度大于等于 512B, 且是 16 的倍数;

底层应答:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x32 一级命令字段							
1(CMD_L)	二级命令字段: 校验分区号							
2~3(LEN)	发送数据长度							

(DAT)	无
4(CR1)	状态字节 1
5(CR2)	状态字节 2

●LEN 发送数据长度：0x00(LEN[0])、0x00(LEN[1])，LEN = LEN[0] + (LEN[1]<<8)。

●状态字节(CR1、CR2)根据命令执行情况分为：

1. 校验成功：状态标志位(0xA0、0x00)。
2. 校验失败：状态标志位(CR1, CR2)
 - (1)、(0xB0、0x00)：返回失败；
 - (2)、(0xB0、0x32)：CRC 校验地址被分区保护；
 - (3)、(0xB0、0x33)：CRC 校验地址范围跨分区；
 - (4)、(0xB0、0x34)：CRC 校验地址范围越界（指超出整个 FLASH 大小）；
 - (5)、(0xB0、0x35)：CRC 校验地址不是 16 字节对齐；
 - (6)、(0xB0、0x36)：CRC 校验长度不是 16 的倍数，或者长度小于 512B；
 - (7)、(0xB0、0x38)：CRC 校验失败；

CRC32 模型如下：

CRC-32/MPEG-2 x32+x26+x23+x22+x16+x12+x11+x10+x8+x7+x5+x4+ \

32 ▾

04C11DB7	例如：3D65
FFFFFFFF	例如：FFFF
00000000	例如：0000

输入数据反转 (REFIN) 输出数据反转 (REFOUT)

CRC 软件实现代码如下：

```

u32 CRC32_Calculate(u32* data,u32 len)
{
    u32 crc=0xffffffff, xbit=0,data0=0,i=0,j=0;
    u32 polynomial = 0x04c11db7;
    for(i=0;i<len;i++)
    {
        xbit = 0x80000000;
        data0 = *data++;
        for(j=0;j<32;j++)
        {
            if(crc & 0x80000000)
            {
                crc= (crc<<1)^polynomial;
            }
            else
            {
                crc<<=1;
            }

            if(data0 & xbit)
            {
                crc ^= polynomial;
            }
            xbit >>= 1;
        }
    }
    return crc;
}
    
```

2.2.6. CMD_OPT_RW

该命令用于选项字节读写（包含了读保护等级、FLASH 页写保护、Data0/1 配置、USER 配置）。当配置了分区，BOOT 将读保护级别由 L1 降为 L0，分区封口后的降级只会导致未分区封口的区域全擦，分区封口的区域不会被擦除。

注：BOOT V1.0 版本不允许 MMU 分区封口下读保护降级。

上层指令：

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x40 一级命令字段							
1(CMD_L)	二级命令字段							
2~3(LEN)	发送数据长度							
4~7(Par)	保留 (0)							
8~21(DAT)	选项字节配置 14 个字节							

●CMD_L 二级命令字段：

1. 0x00：获取选项字节。

2. 0x01: 配置选项字节。
3. 0x02: 配置选项字节, 再复位。

●LEN 发送数据长度: $0x0E(LEN[0])$ 、 $0x00(LEN[1])$, $LEN = LEN[0] + (LEN[1] \ll 8)$ 。

●DAT[0~13]: 选项字节配置 14 个字节

RDP、USER1、USER2、USER3、USER4、USER5、USER6、Data0、Data1、WRP0、WRP1、WRP2、WRP3、RDP2;

1. 当 $CMD_L = 0x00$ 时: DAT[0~13]全部为 0x00。
2. 当 $CMD_L = 0x01/0x02$ 时: DAT[0~13]为配置选项字节要写入的值。

底层应答:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x40 一级命令字段							
1(CMD_L)	二级命令字段							
2~3(LEN)	发送数据长度							
4~17(DAT)	选项字节配置 14 个字节							
18(CR1)	状态字节 1							
19(CR2)	状态字节 2							

●LEN 发送数据长度: $0x0E(LEN[0])$ 、 $0x00(LEN[1])$, $LEN = LEN[0] + (LEN[1] \ll 8)$ 。

●DAT[0~13]: 当前选项字节配置 14 个字节

RDP、USER1、USER2、USER3、USER4、USER5、USER6、Data0、Data1、WRP0、WRP1、WRP2、WRP3、RDP2;

●状态字节(CR1、CR2)根据命令执行情况分为:

1. 返回成功: 状态标志位(0xA0、0x00)。
2. 校验失败: 状态标志位(CR1, CR2)
 - (1)、(0xB0、0x00): 返回失败;
 - (2)、(0xB0、0x39): 已配分区, 不允许读保护级别由 L1 降为 L0;

2.2.7. CMD_USERX_OP

该命令用于读取或者配置分区 USER1/2/3 大小, 分区配置完成后对应的分区自动使能封口, 分区 USER1/2/3 大小只能配置一次。

建议用户的配置流程:

1.如果需要分两个区,需要配置 USER3 大小和 USER2=0KB (配置完自动封口)。如果需要对 USER1 也封口,再配置一下 USER1。USER1 + USER2+USER3 的大小必须为整个 FLASH 的大小;

2.如果需要分三个区,先配置 USER3 (配置完自动封口),再配置 USER2 (配置完自动封口)即可。如果需要对 USER1 也封口,再配置一下 USER1。USER1 + USER2 + USER3 的大小必须为整个 FLASH 的大小。

配置分区操作见下表:

情况描述	分区设置顺序	涉及权限管理的 FLASH 空间大小 ⁽¹⁾			说明
		user1	user2	user3	
没有分区,默认为 user1 区域不封口	-	-	-	-	user1 区域大小为 flash_size, 没有访问权限管理功能
没有分区, USER1 分区封口	user1	user1_size	-	-	user1 区域大小为 flash_size
两个分区, USER1 不封口、USER3 分区封口	user3→user2	-	0	user3_size	user1、user3 空间大小和为 flash_size, user1 没有访问权限管理功能
两个分区, USER1 封口、USER3 分区封口	user3→user2→user1	user1_size	0	user3_size	user1、user3 空间大小和为 flash_size
三个分区, USER1 不封口、USER2&USER3 分区封口	user3→user2	-	user2_size	user3_size	user1、user2 和 user3 空间大小和为 flash_size, user1 没有访问权限管理功能
三个分区, 都分区封口	user3→user2→user1	user1_size	user2_size	user3_size	user1、user2 和 user3 空间大小和为 flash_size
说明:					
1. “涉及权限管理的 FLASH 空间”指的是已设置分区大小的 FLASH 主存储区空间。					

上层指令:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x41 一级命令字段							
1(CMD_L)	二级命令字段							
2~3(LEN)	发送数据长度							
4~7(Par)	Par[0]: 分区 USER1/2/3							
	Par [1]: 分区 USER1/2/3 大小							
	Par [2]: 保留 (0)							
	Par [3]: 保留 (0)							
DAT	无							

●CMD_L 二级命令字段:

1. 0x00: 读取分区 USER1/2/3 大小配置。
2. 0x01: 配置分区 USER1/2/3 大小。
3. 0x02: FLASH (main flash 和 data flash) 封口。

注: CMD_L = 0x02, Par[0~3] 都保留, 可以全写 0; FLASH 封口非分区封口, 指禁用 BOOT 对 main flash 和 data flash 区域的写擦操作。

●LEN 发送数据长度: 0x00(LEN[0])、0x00(LEN[1]), LEN = LEN[0] + (LEN[1]<<8)。

●Par[0]: 分区号

1. 0x00: 分区 USER1。
2. 0x01: 分区 USER2。
3. 0x02: 分区 USER3。

●Par [1]:

1. CMD_L = 0x00: 0x00。
2. CMD_L = 0x01: 分区 USER1/2/3 大小配置
 USER1 分区大小的输入范围: 0x0: 4KB 0x1: 8KB 0x1F: 128KB (default),
 USER2 分区大小的输入范围: 0x0: 0KB(default) 0x1: 4KB... .. 0x1E: 120KB,
 USER3 分区大小的输入范围: 0x0: 0KB(default) 0x1: 4KB... .. 0x1F: 124KB,
 USER1 + USER2 + USER3 = 128KB; 用户区 USER1/2/3 大小配置后自动封口。

分区大小和地址确定

分区的起始地址确定为 0x0800_0000, 分区的末地址为起始地址加整个 FLASH 的

容量（比如 FLASH 的容量为 128K，则末地址为 $0x0800_0000 + 32*0x1000 - 1 = 0x0801_FFFF$ ）。

如果 USER1 分区了，则 USER1 的分区地址范围为 $0x0800_0000 \sim (0x0800_0000 + USER1_Size*0x1000 - 1)$ 。

如果 USER3 分区了，则 USER3 的分区地址范围为 $(0x0802_0000 - USER3_Size*0x1000) \sim 0x0801_FFFF$ （例如 FLASH 末地址为 $0x0801_FFFF$ ）。

USER2 的分区地址，首地址为 USER1 的末地址，末地址为 USER3 的首地址。如果 USER1 没有分区，则 USER2 的首地址需要由 USER2_Size 确定

●Par [2]: 保留 (0)

●Par [3]: 保留 (0)

底层应答:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x41 一级命令字段							
1(CMD_L)	二级命令字段							
2~3(LEN)	发送数据长度							
4~7(DAT)	DAT[0]: 分区 USER1/2/3							
	DAT[1]: 分区 USER1/2/3 大小							
	DAT [2]: 封口状态							
	DAT [3]: 0x00							
8(CR1)	状态字节 1							
9(CR2)	状态字节 2							

●LEN 发送数据长度: $0x02(LEN[0])$ 、 $0x00(LEN[1])$, $LEN = LEN[0] + (LEN[1] \ll 8)$ 。

注: $CMD_L = 0x02$, $DAT[0\sim3]$ 无意义

●DAT[0]: 分区号

1. 0x00: 分区 USER1。
2. 0x01: 分区 USER2。
3. 0x02: 分区 USER3。

●DAT[1]: 读取的当前分区 USER1/2/3 大小

USER1 分区大小的输入范围: 0x0: 4KB 0x1: 8KB 0x1F: 128KB (default),

USER2 分区大小的输入范围：0x0: 0KB(default) 0x1: 4KB... .. 0x1E: 120KB

USER3 分区大小的输入范围：0x0: 0KB(default) 0x1: 4KB... .. 0x1F: 124KB,

USER1 + USER2 + USER3 = 128KB;

●DAT [2]:

1. 0x55 (未封口), 0xAA(已封口)

●DAT [3]: 0x00

●状态字节(CR1、CR2)根据命令执行情况分为:

1. 返回成功: 状态标志位(0xA0、0x00)。

2. 返回失败:

(1)、(0xB0、0x00): 返回失败;

(2)、(0xB0、0x3A): 分区大小已经配置, 无法再次配置;

(3)、(0xB0、0x3B): 分区大小配置错误, 必须满足 USER1 + USER2 + USER3 = FLASH 容量;

(4)、(0xB0、0x3C): 分区配置顺序错误, 必须先配置 USER1 或者 USER3;

2.2.8. CMD_SYS_RESET

该命令用于软件复位 BOOT 程序。

上层指令:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x50 一级命令字段							
1(CMD_L)	0x00 二级命令字段							
2~3(LEN)	发送数据长度							
4~7(Par)	保留							
(DAT)	无							

●保留值: 0x00;

底层应答:

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x50 一级命令字段							
1(CMD_L)	0x00 二级命令字段							

2~3(LEN)	发送数据长度
(DAT)	无
4(CR1)	状态字节 1
5(CR2)	状态字节 2

●状态字节(CR1、CR2)根据命令执行情况分为：

1. 返回成功：状态标志位(0xA0、0x00)。
2. 返回失败：状态标志位(0xB0、0x00)。

2.2.9. CMD_APP_GO

该命令用于 BOOT 下载完应用程序到 FLASH 后跳转 USER1 复位程序入口地址 (0x0800_0000) 执行，或者用 BOOT 下载完应用程序到 SRAM 后跳转 SRAM 复位程序入口地址 (入口地址位置根据用户实际下载决定) 执行，SRAM 支持用户使用地址范围 0x20001000~0x2003FFF。

USER1 分区封口无法执行跳转到 Flash。

上层指令：

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x51 一级命令字段							
1(CMD_L)	0x00 二级命令字段							
2~3(LEN)	发送数据长度							
4~7(Par)	保留							
(DAT)	无							

●CMD_L：跳转分区号

1. 0x00 = Flash;
2. 0x01~0x03, 保留
3. 0x04 = SRAM; 支持地址范围 0x20001000~0x2003FFF。

●LEN 发送数据长度：0x00(LEN[0])、0x00(LEN[1])， $LEN = LEN[0] + (LEN[1] \ll 8)$

●Par[0~3]：跳转的起始地址，合成规则为 $Address = Par[0] | Par[1] \ll 8 | Par[2] \ll 16 | Par[3] \ll 24$ 。

●保留值：0x00;

底层应答：

byte \ bit	b7	b6	b5	b4	b3	b2	b1	b0
0(CMD_H)	0x51 一级命令字段							
1(CMD_L)	0x00 二级命令字段							
2~3 (LEN)	发送数据长度							
(DAT)	无							
4(CR1)	状态字节 1							
5(CR2)	状态字节 2							

●状态字节(CR1、CR2)根据命令执行情况分为：

1. 返回成功：状态标志位(0xA0、0x00)。
2. 返回失败：状态标志位(0xB0、0x00)。

2.3. 返回状态字说明

2.3.1. 返回成功状态字

返回成功：状态标志位(0xA0、0x00)。表示上层下发的命令执行成功，返回成功状态字。

包含了读取、更新、配置等命令的成功返回值。

2.3.2. 返回失败状态字

返回失败：状态标志位(0xB0、0x00)。表示上层下发的命令由于其他原因（命令接受格式错误或者超时等）执行失败，返回失败状态字。

2.3.3. 返回其他状态字

下列的返回状态字也是返回失败，第二字节的状态字表示不同的错误类型。

- (1)、(0xB0、0x30)：在“擦除/下载”时 FLASH 页被 RDP 保护；
- (2)、(0xB0、0x31)：擦除/下载 FLASH 页被 WRP 保护；
- (3)、(0xB0、0x32)：擦除/下载/CRC 校验地址被分区保护；
- (4)、(0xB0、0x33)：擦除/下载/CRC 校验地址范围跨分区；
- (5)、(0xB0、0x34)：擦除/下载/CRC 校验地址范围越界（指超出整个 FLASH/SRAM 大

小);

(6)、(0xB0、0x35): 擦除/下载/CRC 校验起始地址不是 16 字节对齐;

(7)、(0xB0、0x36): 下载/CRC 校验数据长度不是 16 的倍数; 数据长度表示擦除 FLASH/SRAM 的长度, 或者是下载代码到 FLASH/SRAM 的长度, 或者是校验 FLASH/SRAM CRC 值的长度; 代码中是 CRC 校验长度小于 512B;

(8)、(0xB0、0x37): 擦除/下载 FLASH/SRAM 编程失败;

(9)、(0xB0、0x38): CRC 校验失败;

(10)、(0xB0、0x39): 已配分区, 不允许读保护级别由 L1 降为 L0;

(11)、(0xB0、0x3A): 分区已经配置, 无法再次配置;

(12)、(0xB0、0x3B): 分区大小配置错误, 必须满足 $USER1 + USER2 + USER3 = FLASH$ 容量;

(13)、(0xB0、0x3C): 分区配置顺序错误, 必须先配置 USER1 或者 USER3;

(14)、(0xB0、0x42): FLASH 封口错误, FLASH 已封口, 擦除/下载 FLASH 失败。

(15)、(0xB0、0x43): BOOT 上电自校验错误。

(16)、(0xBB、0xCC): 上层发送命令一级、二级命令字段不属于任何命令。

3. BOOT使用说明

3.1. 上位机控制流程

上位机支持用户擦除 FLASH 区，用户代码下载，下载代码完整性校验。

上位机用明文下载用户代码。

上位机支持用户读取和配置分区 USER1/2/3 大小。当用户配置分区大小后不能再修改。

上位机支持用户更新选项字节读取和修改。

上位机支持软件复位命令和跳转 USER1/SRAM 复位程序入口地址执行命令。

进 BOOT：进入 BOOT，此时可以与 PC TOOL 通过 UART1 接口交互；

芯片固件完整性校验：选择从系统存储区启动，BOOT 自动进行完整性自校验，校验失败时会进入死循环，后续的功能无法使用；

命令集交互：PC TOOL 依据 BOOT 支持的命令集发送不同的命令来使用相应的功能；

1. 读取 BOOT 版本号、芯片型号索引、芯片 ID；
2. 擦除 FLASH；
3. 下载用户程序到 FLASH；
4. CRC 校验下载的用户程序；
5. 读取/配置选项字节（包含了读保护等级、FLASH 页写保护、Data0/1 配置、USER 配置）；
6. 获取分区 USERX 大小，配置分区 USERX 大小；
7. 系统复位，可以复位 BOOT 程序重新运行；
8. 跳转 USER1/SRAM 复位程序入口地址，跳转到下载到 USER1/SRAM 分区代码的复位程序入口地址

4. 历史版本

版本	修订日期	说明
V1.0.0	2024/5/14	初始版本
V1.0.1	2024/6/11	1. 增加分区封口操作表格 2. 增加 FLASH 封口描述 3. 增加 L1 降级操作描述
V1.0.2	2026/1/6	1. 修改 CMDDATA_CRC_CECK 实例 2. 更新页眉/页脚

5. 声明

国民技术股份有限公司（下称“国民技术”）对此文档拥有专属产权。依据中华人民共和国的法律、条约以及世界其他法域相适用的管辖，此文档及其中描述的国民技术产品（下称“产品”）为公司所有。

国民技术在此并未授予专利权、著作权、商标权或其他任何知识产权许可。所提到或引用的第三方名称或品牌（如有）仅用作区别之目的。

国民技术保留随时变更、订正、增强、修改和改良此文档的权利，恕不另行通知。请使用者在下单购买前联系国民技术获取此文档的最新版本。

国民技术竭力提供准确可信的资讯，但即便如此，并不推定国民技术对此文档准确性和可靠性承担责任。

使用此文档信息以及生成产品时，使用者应当进行合理的设计、编程并测试其功能性和安全性，国民技术不对任何因使用此文档或本产品而产生的任何直接、间接、意外、特殊、惩罚性或衍生性损害结果承担责任。

国民技术对于产品在系统或设备中的应用效果没有任何故意或保证，如有任何应用在其发生操作不当或故障情况下，有可能致使人员伤亡、人身伤害或严重财产损失，则此类应用被视为“不安全使用”。

不安全使用包括但不限于：外科手术设备、原子能控制仪器、飞机或宇宙飞船仪器、所有类型的安全装置以及其他旨在支持或维持生命的应用。

所有不安全使用的风险应由使用者承担，同时使用者应使国民技术免于因为这类不安全使用而导致被诉、支付费用、发生损害或承担责任时的赔偿。

对于此文档和产品的任何明示、默示之保证，包括但不限于适销性、特定用途适用性和不侵权的保证责任，国民技术可在法律允许范围内进行免责。

未经明确许可，任何人不得以任何理由对此文档的全部或部分进行使用、复制、修改、抄录和传播。