

# 32 位微控制器

## FLASH 操作说明及注意事项

### 适用对象

系列	产品型号	系列	产品型号	系列	产品型号
<b>HC32L110</b>	HC32L110C6UA HC32L110C6PA HC32L110C4UA HC32L110C4PA HC32L110B6PA HC32L110B4PA HC32L110B6YA	<b>HC32F00</b>	HC32F003C4UA HC32F003C4PA HC32F005C6UA HC32F005C6PA HC32F005D6UA	<b>HC32L13</b>	HC32L130E8PA HC32L130F8UA HC32L130J8TA HC32L136J8TA HC32L136K8TA
<b>HC32F03</b>	HC32F030E8PA HC32F030F8UA HC32F030F8TA HC32F030H8TA HC32F030J8TA HC32F030K8TA	<b>HC32L07</b>	HC32L072PATA HC32L072KATA HC32L072JATA HC32L073PATA HC32L073KATA HC32L073JATA	<b>HC32F07</b>	HC32F072PATA HC32F072KATA HC32F072JATA
<b>HC32L17</b>	HC32L176PATA HC32L176MATA HC32L176KATA HC32L176JATA HC32L170JATA HC32L170FAUA	<b>HC32F17</b>	HC32F176PATA HC32F176MATA HC32F176KATA HC32F176JATA HC32F170JATA HC32F170FAUA	<b>HC32L19</b>	HC32L196PCTA HC32L196MCTA HC32L196KCTA HC32L196JCTA HC32L190JCTA HC32L190FCUA
<b>HC32F19</b>	HC32F196PCTA HC32F196MCTA HC32F196KCTA HC32F196JCTA HC32F190JCTA HC32F190FCUA				

# 目 录

<b>1</b>	<b>摘要 .....</b>	<b>3</b>
<b>2</b>	<b>FLASH 简介 .....</b>	<b>3</b>
<b>3</b>	<b>FLASH 应用说明 .....</b>	<b>4</b>
3.1	简介 .....	4
3.2	安全特性 .....	4
3.2.1	操作源保护 .....	4
3.2.2	操作目标保护 .....	4
3.2.3	PC 地址擦写保护 .....	4
3.2.4	寄存器写保护 .....	5
3.3	功能说明 .....	5
3.4	工作流程介绍 .....	6
3.4.1	Sector 擦除 .....	6
3.4.2	Chip 擦除 .....	6
3.4.3	写操作 .....	7
3.4.4	读操作 .....	7
3.5	基于 FLASH 安全特性的编程方法 .....	8
3.5.1	基于 Keil MDK 的编程方法 .....	8
3.5.2	基于 IAR 的编程方法 .....	8
3.5.3	结果查看与示例 .....	8
<b>4</b>	<b>总结 .....</b>	<b>9</b>
<b>5</b>	<b>版本信息 &amp; 联系方式 .....</b>	<b>10</b>

## 1 摘要

本篇应用笔记主要介绍华大半导体 MCU\*的 FLASH 安全特性、操作说明及应用方法。

注意：

- 本应用笔记为华大半导体 MCU\*的应用补充材料，不能代替用户手册，具体功能及寄存器的操作等相关事项请以用户手册为准。

## 2 FLASH 简介

### 什么是 FLASH？

闪存器件的一种，闪存是一种非易失性（Non-Volatile）内存，在没有电流供应的条件下也能够长久地保持数据，其存储特性相当于硬盘，这项特性正是闪存得以成为各类便携型数字设备的存储介质的基础。

（引自‘百度百科’，‘互动百科’，‘维基百科’）

### FLASH 特点？

Flash 闪存是非易失存储器，可以对称为块的存储器单元块进行擦写和再编程。任何 Flash 器件的写入操作只能在空或已擦除的单元内进行，所以大多数情况下，在进行写入操作之前必须先执行擦除。

### FLASH 的应用？

FLASH 被广泛用于移动存储、MP3 播放器、数码相机、掌上电脑等新兴数字设备中。

**\*支持型号见封面。**

## 3 FLASH 应用说明

### 3.1 简介

华大半导体 MCU\*根据型号不同，涵盖了 16/32/64/128/256/512K 字节（Byte）容量的 FLASH 存储器，每块 FLASH 按照 Sector 进行划分，每个 Sector 容量为 512 字节。本存储器支持对擦除（片/页擦除）、编程以及读取操作。此外，本模块还支持对 FLASH 存储器擦写的保护，以及控制寄存器的写保护。

### 3.2 安全特性

#### 3.2.1 操作源保护

华大半导体 MCU\*对容量超过 32K 的 FLASH 采用高安全性的硬件设计，具有 FLASH 操作源防御功能：仅当 FLASH 操作函数的地址位于 0~32K，才能正确执行 FLASH 擦写操作。

FLASH 地址的 0~32K 具有更高的安全性，重要函数须放置此区域内，例如重要程序入口、中断入口函数、高安全算法模块、UID、AES、真随机数、RTC 的算法配合，组成高安全认证系统。

#### 3.2.2 操作目标保护

整个 64K 字节 FLASH 存储器被划分为 128 个页，每 4 个页共用一个擦写保护位。当页被保护时，对该页进行的擦写操作均无效，并产生报警标志位和中断信号。当 FLASH 存储器中的任意页被保护时，对该 FLASH 的全片擦写无效，并产生报警标志位和中断信号。

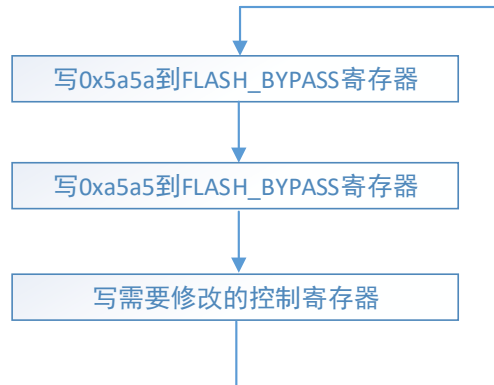
#### 3.2.3 PC 地址擦写保护

CPU 在 FLASH 中运行程序时，如果当前 PC 指针正好落在待擦写的页地址范围之内，那么该擦写操作无效并产生报警标志位和中断信号。

**\*支持型号见封面。**

### 3.2.4 寄存器写保护

本模块的控制器屏蔽普通的写操作，必须用写序列方式才能修改。具体操作步骤如下图所示：



注意：

- 写 0x5a5a、0xa5a5、写目标寄存器，这三步写操作之间不可插入任何写操作（写 ROM、RAM、REG），否则无法改写目标寄存器的数值。如改写失败，需要重新进行这三步操作。

## 3.3 功能说明

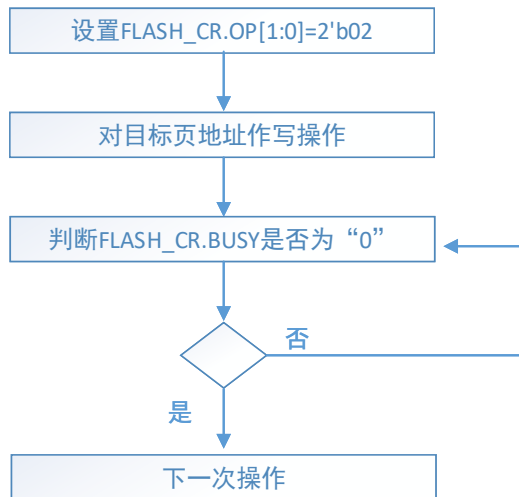
本小节介绍 FLASH 控制器模块功能、工作流程及基于安全特性的编程方法。

本 FLASH 控制器支持对 eFLASH 的 Byte（8bits）、Half-word（16bits）、Word（32bits）三种位宽读写操作。注意，Byte 操作的地址必须按 Byte 对齐，Half-word 操作的目标地址必须按 Half-word 对齐（地址最低位为 1'b0），Word 操作的地址必须按 Word 对齐（地址最低两位为 2'b00）。如果读写操作的地址没有按照位宽规定对齐，该操作无效，并且系统会进入 Hard Fault 出错中断。

## 3.4 工作流程介绍

详细操作步骤，请参考相应系列的用户手册。

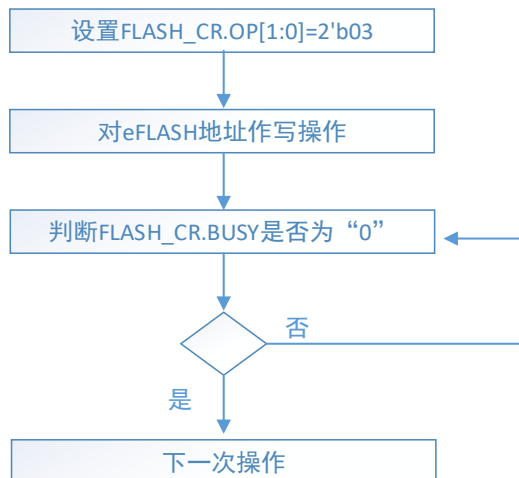
### 3.4.1 Sector 擦除



注意：

1. 控制器忽略目标地址的低9位，只要目标地址落在该页地址范围之内即可。
2. 该写操作用于触发页擦除操作，所写数据也会被控制器忽略。
3. 如果当前擦除指令是在eFLASH内执行，则CPU取值会停止，硬件自动等待eFLASH的BUSY状态结束
4. 如果当前擦除指令是在RAM内执行，则CPU取值不会停止，在对eFLASH进行任何操作之前，软件必须判断eFLASH的BUSY状态是否结束

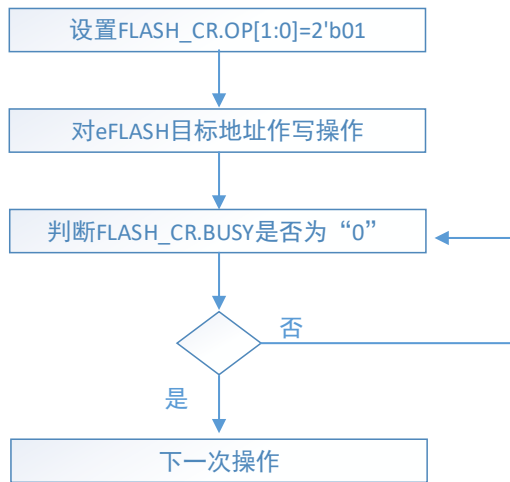
### 3.4.2 Chip 擦除



注意：

1. 控制器忽略目标地址的低15位，只要目标地址落在eFLASH地址范围之内即可。
2. 该写操作用于触发页擦除操作，所写数据也会被控制器忽略。
3. 如果当前擦除指令是在eFLASH内执行，则CPU取值会停止，硬件自动等待eFLASH的BUSY状态结束
4. 如果当前擦除指令是在RAM内执行，则CPU取值不会停止，在对eFLASH进行任何操作之前，软件必须判断eFLASH的BUSY状态是否结束

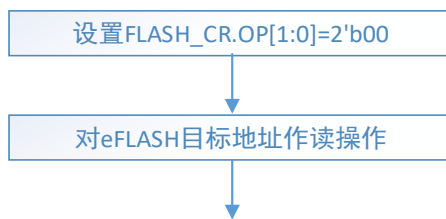
### 3.4.3 写操作



注意:

1. 如果当前擦除指令是在eFLASH内执行, 则CPU取值会停止, 硬件自动等待eFLASH的BUSY状态结束
2. 如果当前擦除指令是在RAM内执行, 则CPU取值不会停止, 在对eFLASH进行任何操作之前, 软件必须判断eFLASH的BUSY状态是否结束

### 3.4.4 读操作



注意:

1. 第一步设置FLASH\_CR.OP[1:0]的步骤实际可以省略, 无论FLASH\_CR.OP[1:0]为何值, 读操作都可以进行

## 3.5 基于 FLASH 安全特性的编程方法

在实际应用中，对于容量大于 32K 的 MCU，如果需要将 FLASH 的操作函数、安全功能函数等放置在 FLASH 的 32K 安全区，可通过以下便捷的方式实现。

说明：

- 本例为说明需要，主要示例将函数“Flash\_SectorErase()”放置在安全区“0x400”地址的方法，实际应用当中可以将“示例函数”和“地址”根据自己的需求进行替换。

### 3.5.1 基于 Keil MDK 的编程方法

在 Keil MDK 中，可以简单通过如下方式实现对安全函数的执行地址映射：

在目标函数的声明处增加以下代码：

```
en_result_t Flash_SectorErase(uint32_t u32SectorAddr) __attribute__((section(".ARM.__at_0x400")));
```

### 3.5.2 基于 IAR 的编程方法

1、在目标函数定义处增加以下代码：

```
en_result_t Flash_SectorErase(uint32_t u32SectorAddr) @".Flash_SectorErase"
```

2、在工程“\*.icf”文件中增加以下代码：

```
place at address mem:0x00000400 { readonly section .Flash_SectorErase};
```

### 3.5.3 结果查看与示例

如果对此方法产生的具体结果信息感兴趣，可以通过 map 文件、调试文件或在调试运行期间观测代码的执行地址空间。

示例如下（可以确认目标函数确实放在了预期[0x400]地址）：

Name	Address	Size	#Insts	Source
* * *	*	*	*	*
Flash_Init	0000 0180	156	71	flash.c:229
Flash_LockAll	0000 0214	48	24	flash.c:627
FLASH_RAM_IRQHandler	0000 0174	12	6	interrupts_b
<b>Flash_SectorErase</b>	<b>0000 0400</b>	<b>144</b>	<b>64</b>	<b>flash.c:499</b>
Flash_UnlockAll	0000 0238	48	22	flash.c:647
Flash_WriteByte	0000 0260	144	70	flash.c:287
HardFault_Handler	0000 02E4	16	8	interrupts_b
HardFault_Handler	0000 00F0	2	1	sysctrl.c:19
I2C0_IRQHandler	0000 02F4	8	4	interrupts_b
I2C1_IRQHandler	0000 02FC	8	4	interrupts_b
LCD_IRQHandler	0000 0304	8	4	interrupts_b
LPTIM0_1_IRQHandler	0000 030C	12	6	interrupts_b
LPUART0_IRQHandler	0000 0318	8	4	interrupts_b

```

496  ** \retval ErrorInvalidParameter FLASH地址无效
497  ** \retval ErrorTimeout 操作超时
498  ****
499  en_result_t Flash_SectorErase(uint32_t u32SectorAddr)
500  {
501      0000 0400 PUSH (R3-R7, LR)
502      0000 0402 MOV R4, R0
503      en_result_t enResult = Ok;
504      volatile uint32_t u32TimeOut = FLASH_TIMEOUT_ERASE;
505      0000 0404 MOVS R0, #0xFF
506      0000 0406 STR R0, [SP]
507      0000 0408 MOVS R0, #4
508  }

```



## 4 总结

以上章节主要介绍了华大半导体 MCU\*的 FLASH 的安全特性、工作流程和基于 FLASH 安全特性的编程方法，在实际开发中用户可以根据以上方法进行修改或扩展来满足自己的应用。

\*支持型号见封面。

## 5 版本信息 & 联系方式

日期	版本	修改记录
2019/8/19	Rev1.0	初版发布



---

如果您在购买与使用过程中有任何意见或建议，请随时与我们联系。

Email: [mcu@hdsc.com.cn](mailto:mcu@hdsc.com.cn)

网址: [www.hdsc.com.cn](http://www.hdsc.com.cn)

通信地址: 上海市张江高科园区碧波路 572 弄 39 号

邮编: 201203

---

