32 位微控制器

HC32F4A0 系列的嵌入式 FLASH

本产品支持芯片系列如下

F 系列	**HC32F4A0**

目　录

1 摘要

本篇应用笔记主要介绍如何使用 HC32F4A0 系列芯片的嵌入式 FLASH 读写数据。

2 FLASH 简介

FLASH 接口通过 FLASH ICODE，DCODE 和 MCODE 总线对 FLASH 进行访问。该接口可对 FLASH 执行编程，擦除和全擦除操作；通过指令预取和缓存机制加速代码执行。

HC32F4A0 系列有 2Mbytes 和 1Mbytes 两种不同大小的 FLASH，FLASH 地址结构如图 2-1：

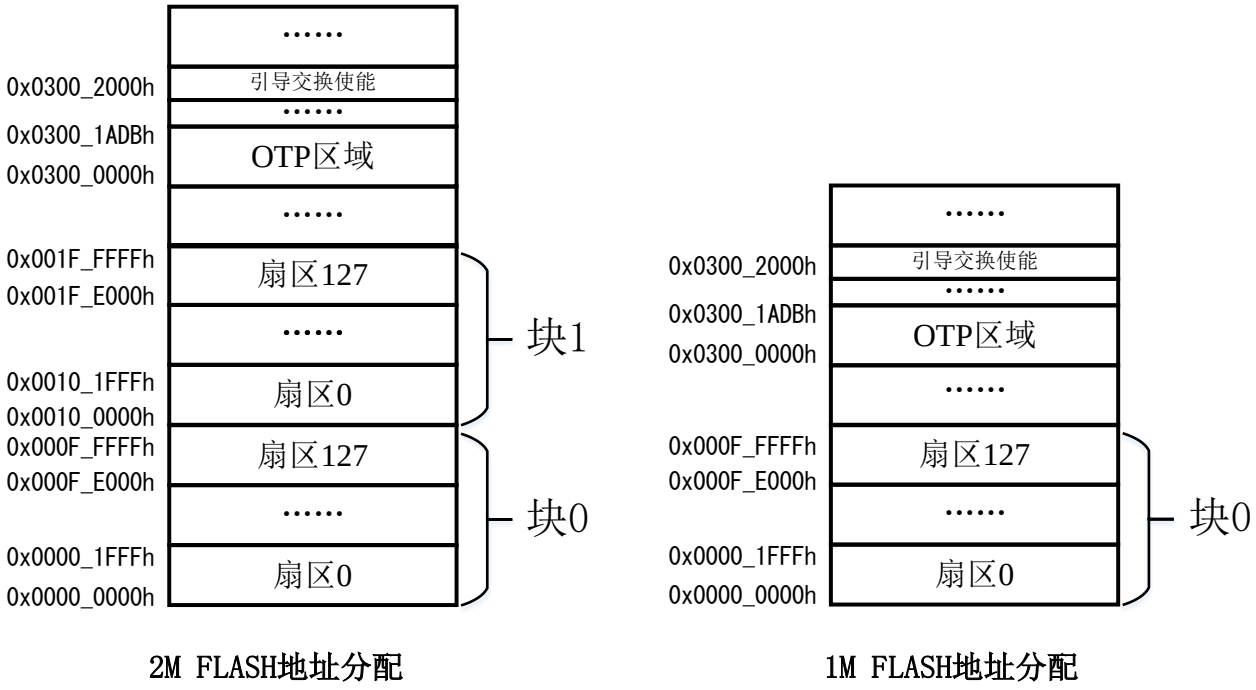

图 2-1 FLASH 地址分配

3 HC32F4A0 系列的 FLASH

3.1 主要特性

- FLASH 读、编程、扇区擦除和全擦除操作。
- 容量最大为 2Mbytes，由两块 1Mbytes 的 FLASH 构成，共 256 个扇区，每个扇区 8Kbytes。块 0 中扇区 0~扇区 15 可配置为 OTP 区域。
- 134Kbytes 的 OTP 空间。
- ICODE 总线 16Bytes 预取指。
- 128 位宽数据读取，读缓存 128 位缓冲，加速代码执行。
- 编程单位为 4bytes，擦除单位为 8Kbytes。
- 两个独立缓存区：ICODE 总线缓存空间 4Kbytes；DCODE 总线缓存空间 1Kbytes。
- 可实现 BGO（Back Ground Operation）功能。
- 支持引导交换功能（1Mbytes 产品没有此功能）。
- 支持安全保护。

3.2 寄存器介绍

EFM_BASE_ADDR：0x40010400

寄存器名	偏移量	功能说明
EFM_FAPRT	0x0000h	FLASH 寄存器保护寄存器。访问 EFM 的寄存器需要先解锁该寄存器
EFM_KEY1	0x0004h	FLASH 密钥 1 寄存器。该寄存器的功能是保护 EFM_FWMC 寄存器
EFM_KEY2	0x0008h	FLASH 密钥 2 寄存器。该寄存器是对 OTP 锁存区域的写保护寄存器
EFM_FSTP	0x0014h	FLASH 停止寄存器。该寄存器控制 FLASH 停止或活动
EFM_FRMC	0x0018h	FLASH 读模式寄存器。设置插入等待周期，缓存功能，预取指令等
EFM_FWMC	0x001Ch	FLASH 擦写模式寄存器。配置编程/擦除模式，总线状态等
EFM_FSR	0x0020h	FLASH 状态寄存器。查看 FLASH 状态，结束标志、错误标志等
EFM_FSCLR	0x0024h	FLASH 状态清除寄存器。清除 FLASH 状态
EFM_FITE	0x0028h	中断许可寄存器。配置 FLASH 中断功能是否使能
EFM_FSWP	0x002Ch	引导交换状态寄存器。查看引导交换功能是否开启
EFM_CHIPID	0x0040h	芯片专属标志寄存器
EFM_UQID0~2	0x0050h	Unique ID 寄存器
EFM_WLOCK	0x0180h	FLASH 写保护锁定寄存器。该寄存器是 EFM_FxNWPRTy 的锁定寄存器
EFM_FxNWPRTy（x=0,1 y=0~3）	0x0190h	FLASH 扇区保护寄存器。寄存器的每一位对应一个扇区的写保护，对扇区操作要先解除扇区对应的写保护

3.3 工作流程介绍

3.3.1 寄存器解除保护和写保护

本模块的寄存器受 EFM_FAPRT 寄存器保护，当处于保护状态，屏蔽普通的写操作。解除保护的步骤如图 3-1：

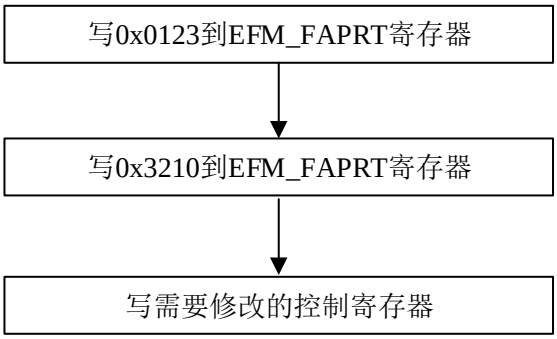

图 3-1 FLASH 寄存器解除保护

在解除保护的状态下，对 EFM_FAPRT 寄存器写任意值，EFM 寄存器再次进入保护状态。

注意：

- 在实际应用中，发现寄存器值未写入成功，应首先检查 EFM 寄存器访问保护是否有效，保护有效时，EFM_FAPRT 寄存器值读出为 0x00000000。

3.3.2 扇区擦除

对 FLASH 进行扇区擦除操作后，该扇区内地址(8KBytes 空间)数据刷新为全 1。扇区擦除操作设定步骤如图 3-2：

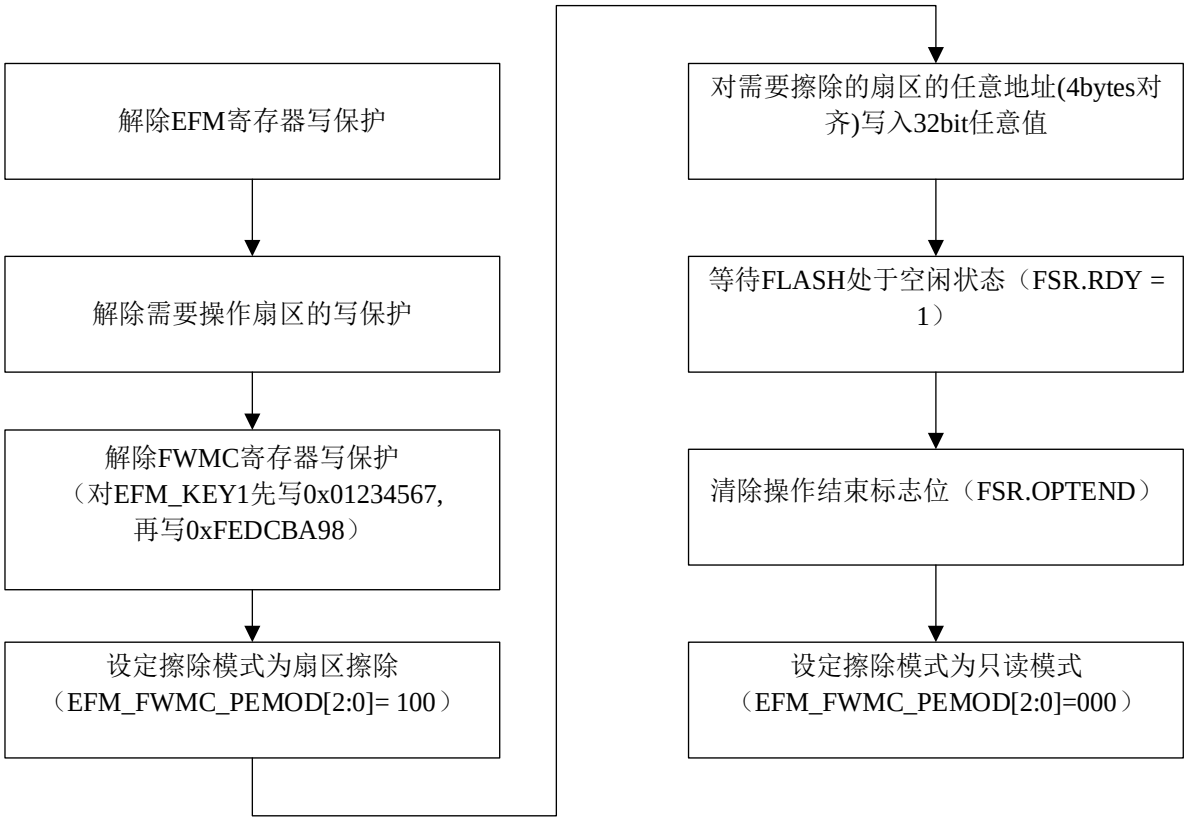

图 3-2 扇区擦除步骤

注意：

1) 对已锁存的 OTP 区域地址无法进行擦除/编程操作。
2) 擦除完成后需把 EFM_FWMC.PEMOD[2:0]设定为只读模式，防止对 FLASH 误操作，导致整个扇区擦除。

3.3.3 全擦除

EFM 提供了单块 FLASH 全擦除和两块 FLASH 同时全擦除两种擦除方式。2MbytesFLASH 的型号可以选择擦除 2 块 FLASH 的任意一块或者两块全擦除。对 FLASH 进行全擦除操作后整个 FLASH 区域所有地址数据刷新为全 1。全擦除的设定步骤如图 3-3：

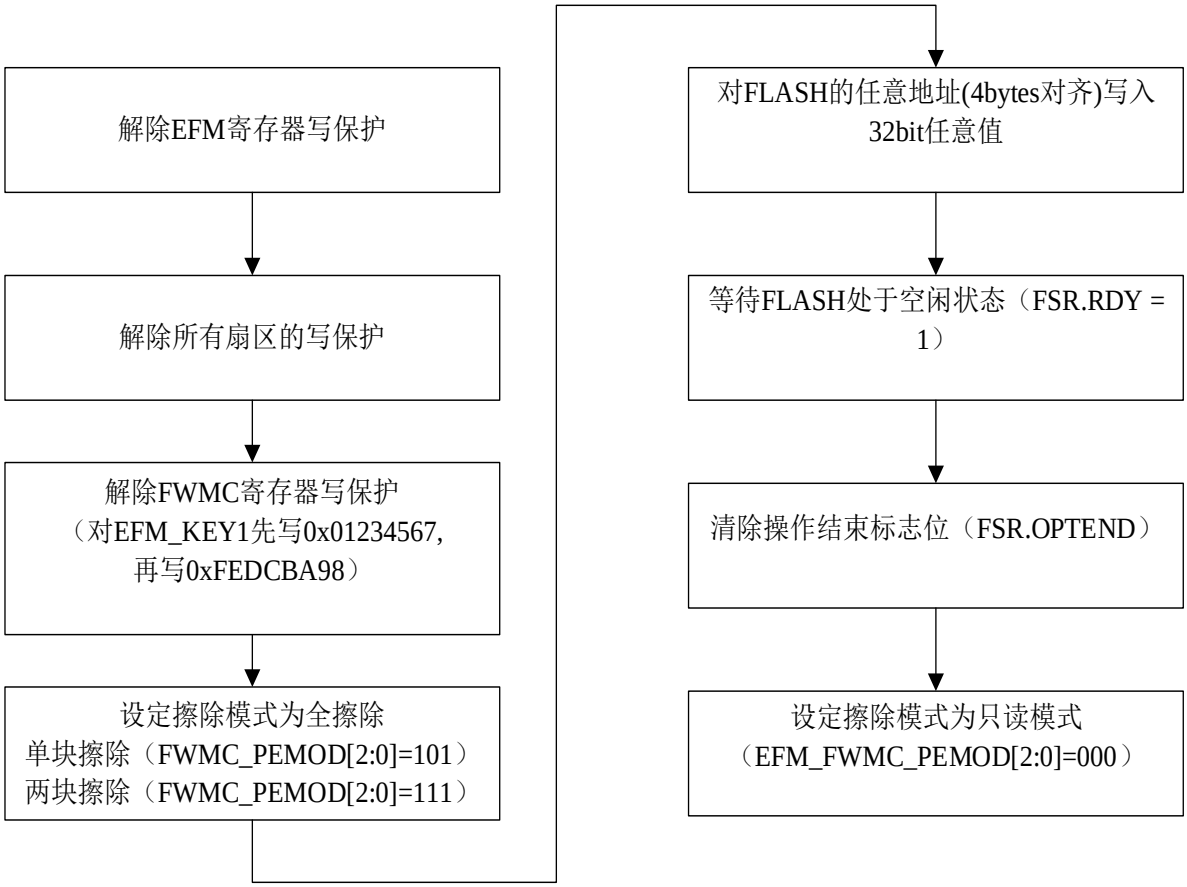

图 3-3 全擦除步骤

注意：

1) 擦除完成后需把 EFM_FWMC.PEMOD[2:0]设定为只读模式，预防对 FLASH 误操作，导致整个 FLASH 擦除。
2) OTP 使能后，对 FLASH 块 0 地址进行单块全擦除操作，全擦除不成功，FLASH 数据保留，标志位 EFM_FSR.OTPWERR0 置位；对 FLASH 块 0 地址发行两块全擦除写操作，FLASH0 数据保留，FLASH1 数据被全擦，标志位 EFM_FSR.OTPWERR0 置位。

3.3.4 单次编程无回读

单次编程无回读模式设定步骤如图 3-4：

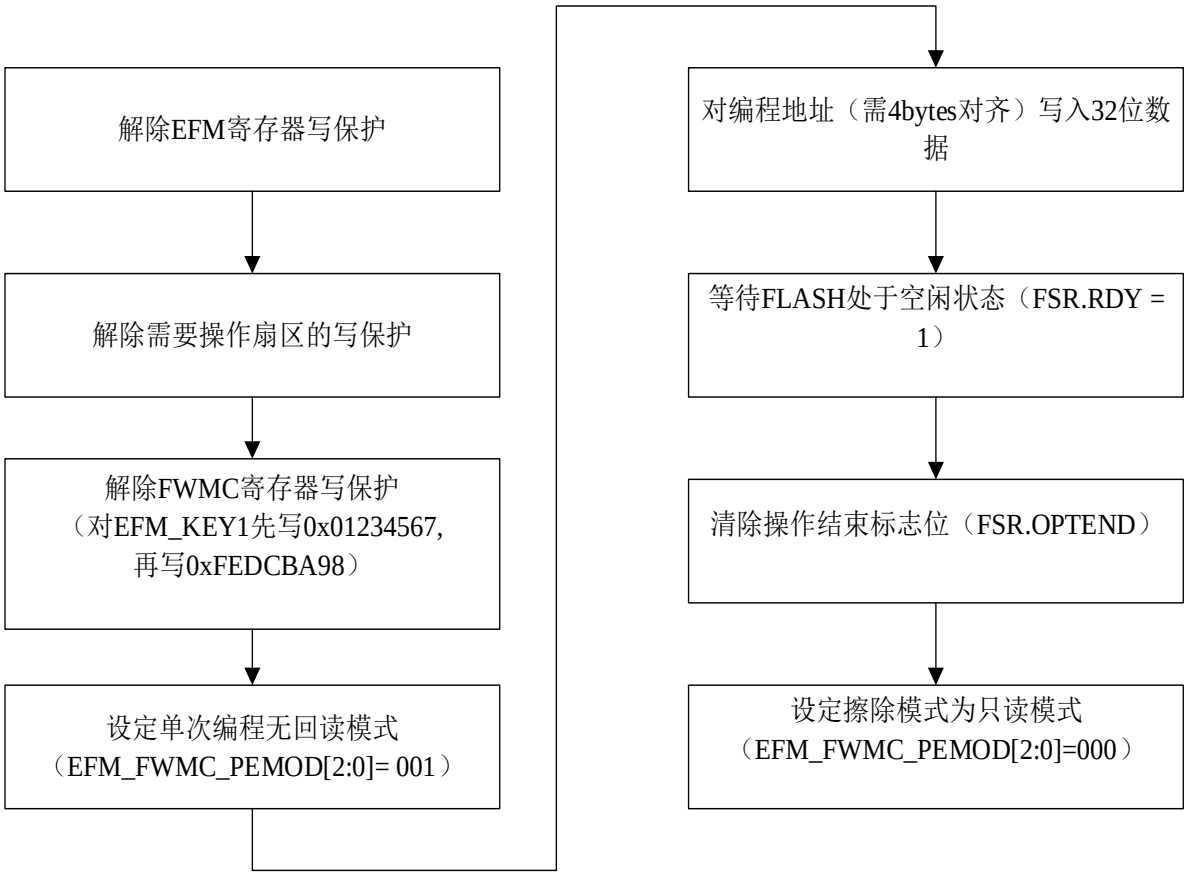

图 3-4 单次编程无回读模式设定步骤

注意：

- 对已锁存的 OTP 地址发行单次编程无回读写操作，编程不成功，标志位 EFM_FSR.OTPWERR0 置位。

3.3.5 单次编程回读

单次编程回读模式是指编程结束后硬件自动读取编程地址数据并和写入数据对比，判断标志位 EFM_FSR.MISMTCH0/1 验证写入数据正确。单次编程回读模式设定步骤如图 3-5：

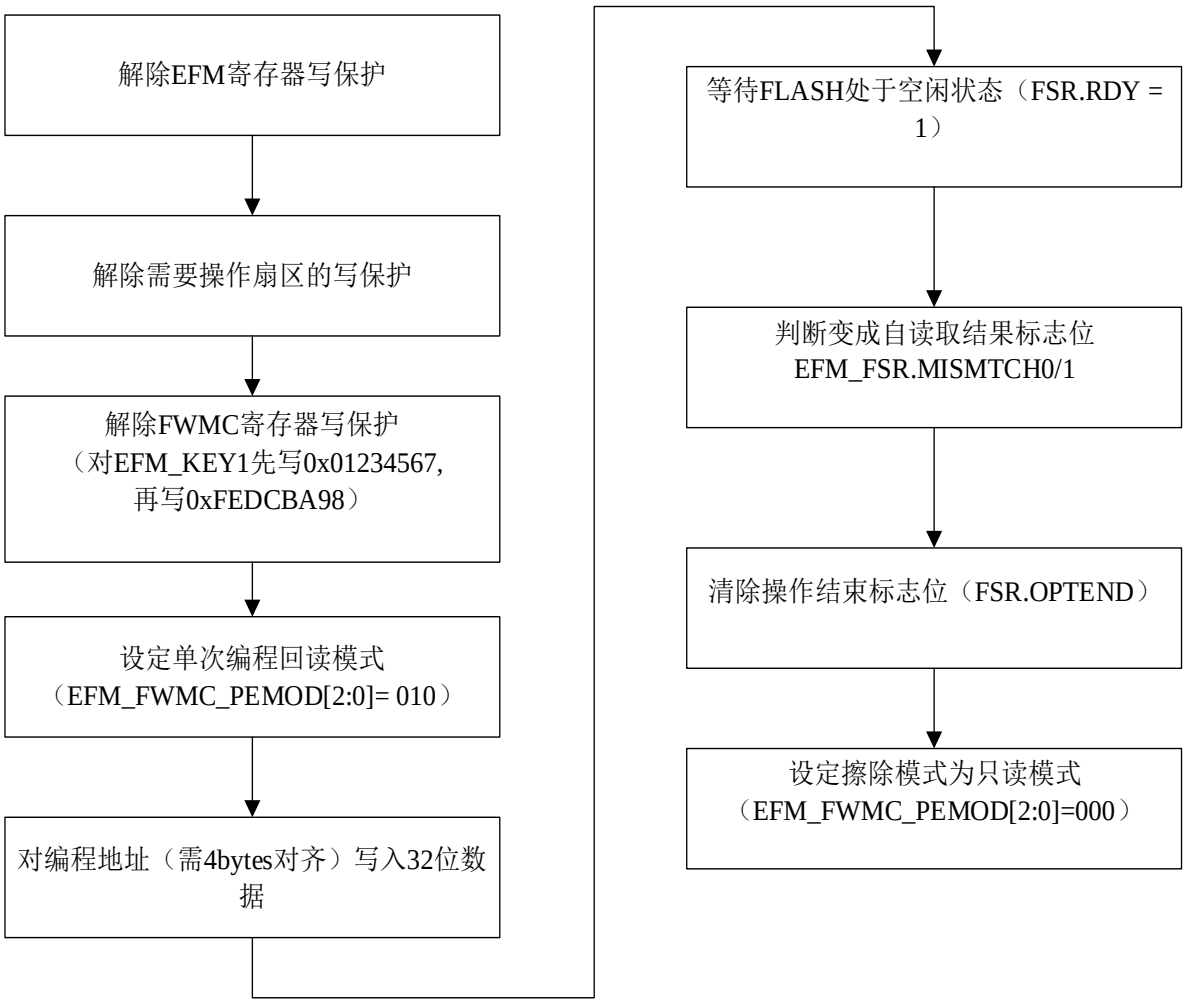

图 3-5 单次编程回读模式步骤

注意：

- EFM_FSR.PGMISMTCH 为 0，表示编程成功，为 1 表示该 FLASH 地址已遭破坏，永久废弃。

3.3.6 连续编程

当连续对 FLASH 地址进行编程时，可以使用连续编程模式。连续编程模式比单编程模式可以节省时间 50%以上。连续编程模式时，对 FLASH 地址写命令间隔不能超过 16us。连续编程操作设定步骤如图 3-6：

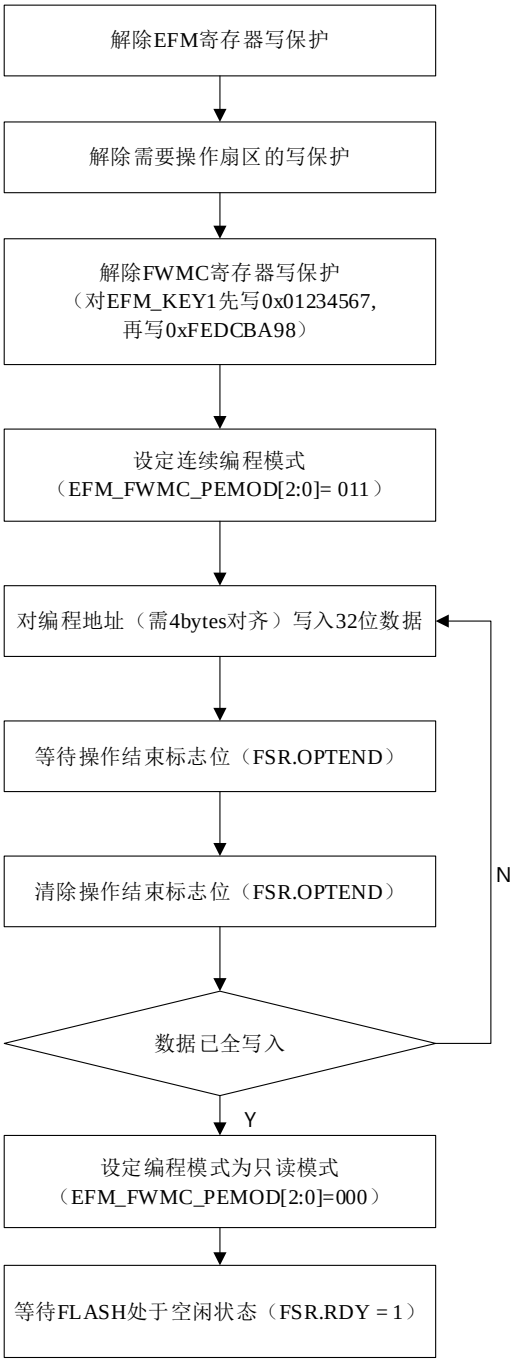

图 3-6 连续编程步骤

连续编程模式下，应用程序不能对其所在的 FLASH 块进行编程操作。例如，需要对 FLASH0 进行连续编程，应用程序应该放在 FLASH1 或者 RAM 里面。

3.4 BGO 功能

BGO（Back Ground Operation）功能是指 FLASH 编程，擦除期间总线处于释放状态。通过设定寄存器 EFM_FWMC.BUSHLDCTL 位，可设定 FLASH 编程、擦除期间，总线处于保持还是释放状态。FLASH 编程、擦除指令在 FLASH 上执行时，该控制位必须设定为 0。擦除指令在 FLASH 以外空间（例如 RAM）执行时，可根据需求自由设定。

3.5 一次性可编程字节（OTP）

本系列 MCU 提供最大 134Kbytes 的 OTP 区域，地址分布如表 3-1：

sector	OTP 块数据地址	容量	OTP 块锁存地址
0	0x0000_0000~0x0000_1FFF	8KB	0x0300_1800~0x0300_1803
1	0x0000_2000~0x0000_3FFF	8KB	0x0300_1804~0x0300_1807
⋮	⋮	⋮	⋮
14	0x0001_C000~0x0001_DFFF	8KB	0x0300_1838~0x0300_183B
15	0x0001_E000~0x0001_FFFF	8KB	0x0300_183C~0x0300_183F
16	0x0300_0000~0x0300_07FF	2KB	0x0300_1840~0x0300_1843
17	0x0300_0800~0x0300_0FFF	2KB	0x0300_1844~0x0300_1847
18	0x0300_1000~0x0300_10FF	256B	0x0300_1848~0x0300_184B
19	0x0300_1100~0x0300_11FF	256B	0x0300_184C~0x0300_184F
20	0x0300_1200~0x0300_12FF	256B	0x0300_1850~0x0300_1853
21	0x0300_1300~0x0300_13FF	256B	0x0300_1854~0x0300_1857
22	0x0300_1400~0x0300_140F	16B	0x0300_1858~0x0300_185B
⋮	⋮	⋮	⋮
53	0x0300_15F0~0x0300_15FF	16B	0x0300_18D4~0x0300_18D7
54	0x0300_1600~0x0300_1603	4B	0x0300_18D8~0x0300_18DB
⋮	⋮	⋮	⋮
181	0x0300_17FC~0x0300_17FF	4B	0x0300_1AD4~0x0300_1AD7

表 3-1 OTP 区域地址分布

OTP 区域未被使能和锁存时，可作为普通 FLASH 空间使用，可多次编程和擦除。一旦 OTP 使能且对应的区域已锁定，则该区域为只读，无法再进行编程和擦除。若是 FLASH 块 0 的 OTP 区域锁定，则 FLASH 块 0 无法进行全擦除。对 OTP 区域操作如图 3-7：

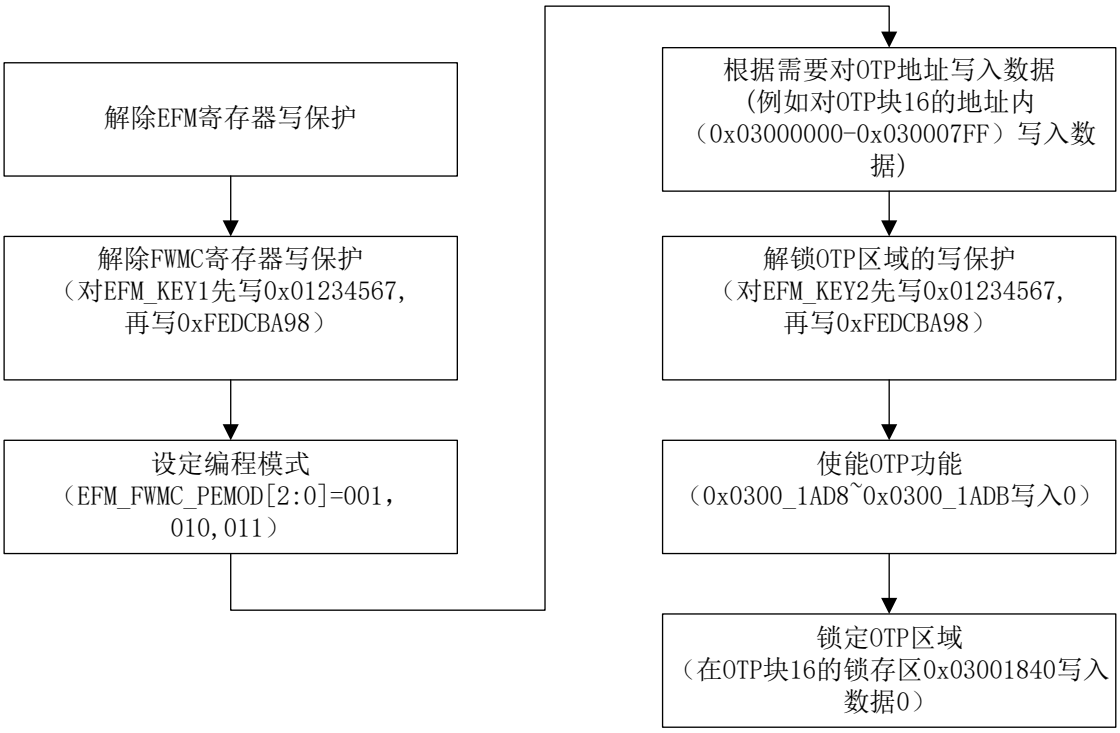

图 3-7 OTP 操作步骤

对已经锁存的 OTP 数据区域地址进行编程，扇区擦除和全擦除，将会产生 OTPWERR0 错误标志。OTP 的具体样例请参考 EFM 模块下的 efm_otp 样例代码。

3.6 引导交换

2MbytesFLASH 的产品，对 FLASH 地址（0x0300_2000）编程写入数据 0x5A5A5A，可开启引导交换功能。具体变换规则如图 3-8、图 3-9 所示：

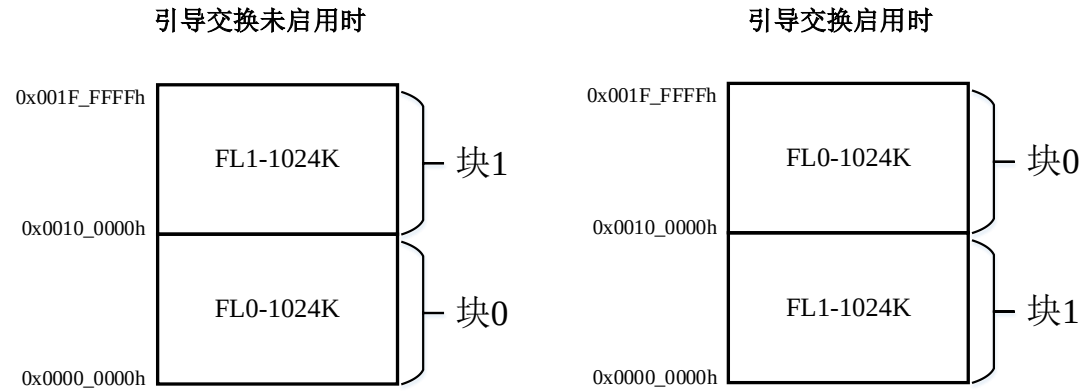

图 3-8 OTP 未启用时，引导交换地址分配

OTP 功能未启用时，对 FLASH 地址（0x0300_2000）编程写入 0x5A5A5A，再进行复位，FLASH 块 0 和 FLASH 块 1 全体地址互换。引导交换标志位 EFM_FSWP.FSWP 会置起，表示引导交换功能启用。此时，OTP 功能无法启用（OTP 使能地址 0x03001AD8 无法编程）。

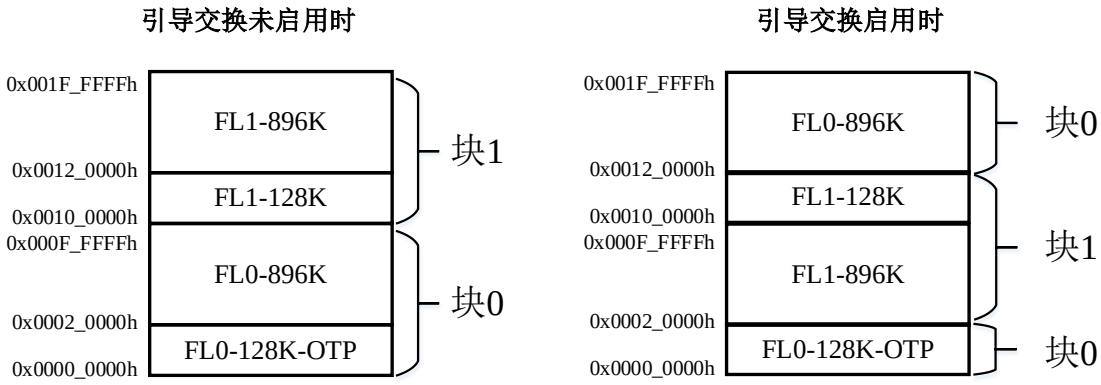

图 3-9 OTP 启用时，引导交换地址分配

OTP 功能启用时，对 FLASH 地址（0x0300_2000）编程写入 0x5A5A5A，再进行复位，FLASH 块 0 和 FLASH 块 1 地址部分互换。此时，引导交换标志位 EFM_FSWP.FSWP 不会置起，需要通过读取地址 0x03002000 的数据来判断引导交换功能是否开启。

4 样例代码

4.1 代码介绍

用户可根据上述的工作流程编写自己的代码来学习验证该模块，也可以直接通过华大半导体的网站下载到设备驱动库（Device Driver Library, DDL）的样例代码并使用其中的 EFM 的样例进行验证。

以下部分简要介绍本应用笔记基于 DDL 的 EFM 模块样例 efm_sequence_program 代码所涉及的各项配置。

1) 解除 FLASH 寄存器保护：

```
/* Unlock peripherals or registers */
Peripheral_WE();
```

2) 设置系统时钟为 240MHZ：

```
/* Configure system clock. */
SystemClockConfig();
```

3) 配置 FLASH 并使能，等待 FLASH 处于空闲状态：

```
/* EFM default config. */
(void)EFM_StructInit(&stcEfmCfg);
/* EFM config */
stcEfmCfg.u32BusStatus  = EFM_BUS_RELEASE;  /* Bus release while programming or erasing */
/*
   * Clock <= 40MHZ           EFM_WAIT_CYCLE_0
   * 40MHZ < Clock <= 80MHZ    EFM_WAIT_CYCLE_1
   * 80MHZ < Clock <= 120MHZ   EFM_WAIT_CYCLE_2
   * 120MHZ < Clock <= 160MHZ  EFM_WAIT_CYCLE_3
   * 160MHZ < Clock <= 200MHZ  EFM_WAIT_CYCLE_4
   * 200MHZ < Clock <= 240MHZ  EFM_WAIT_CYCLE_5
   */
stcEfmCfg.u32WaitCycle = EFM_WAIT_CYCLE_5; /* 5-wait @ 240MHz */
(void)EFM_Init(&stcEfmCfg);
/* Wait flash0, flash1 ready. */
do{
flag1 = EFM_GetFlagStatus(EFM_FLAG_RDY0);
flag2 = EFM_GetFlagStatus(EFM_FLAG_RDY1);
}while((Set != flag1) || (Set != flag2));
```

4) 解锁全部扇区（2MbytesFLASH）：

```
/* FLASH disable write protection (sector 0~255) */
(void)EFM_SectorCmd_Sequential(EFM_ADDR_SECTOR0, 256U, Enable);
```

5) FLASH 块 0 和块 1 分别进行片擦除，然后进行连续编程，编程结束后对两块 FLASH 进行全擦除：

```
/* Chip Erase: flash0 chip erased (sector 0~127)*/
(void)EFM_ChipErase(EFM_MODE_ERASECHIP1, EFM_ADDR_SECTOR0);
/* Sequence program. */
(void)EFM_SequenceProgram(EFM_ADDR_SECTOR0, sizeof(u32TestBuf), u32TestBuf);
/* Chip Erase: flash1 chip erased (sector 128~255)*/
(void)EFM_ChipErase(EFM_MODE_ERASECHIP1, EFM_ADDR_SECTOR128);
/* Sequence program. */
(void)EFM_SequenceProgram(EFM_ADDR_SECTOR128, sizeof(u32TestBuf), u32TestBuf);
/* Chip Erase: flash All erased */
(void)EFM_ChipErase(EFM_MODE_ERASEFULL, EFM_ADDR_SECTOR0);
```

6) 对全部扇区进行锁定并使能 FLASH 寄存器的写保护：

```
/* FLASH enable write protection (sector 0~255) */
(void)EFM_SectorCmd_Sequential(EFM_ADDR_SECTOR0, 256U, Disable);
/* Lock peripherals or registers */
Peripheral_WP();
```

4.2 代码运行

用户可以通过华大半导体的网站下载到 HC32F4A0 的 DDL 的样例代码（efm_sequence_program），并配合评估用板（EV_F4A0_LQ176_V10）运行相关代码学习使用 FLASH 模块。

以下部分主要介绍如何在评估板上运行 FLASH 样例代码并观察结果：

- 确认安装正确的 IAR EWARM v7.7 工具（请从 IAR 官方网站下载相应的安装包，并参考用户手册进行安装）。
- 从华大半导体网站下载 HC32F4A0 DDL 代码。
- 下载并运行 efm\ efm_sequence_program \中的工程文件：

  1) 打开 efm_simple \工程，并打开‘main.c’如下视图：

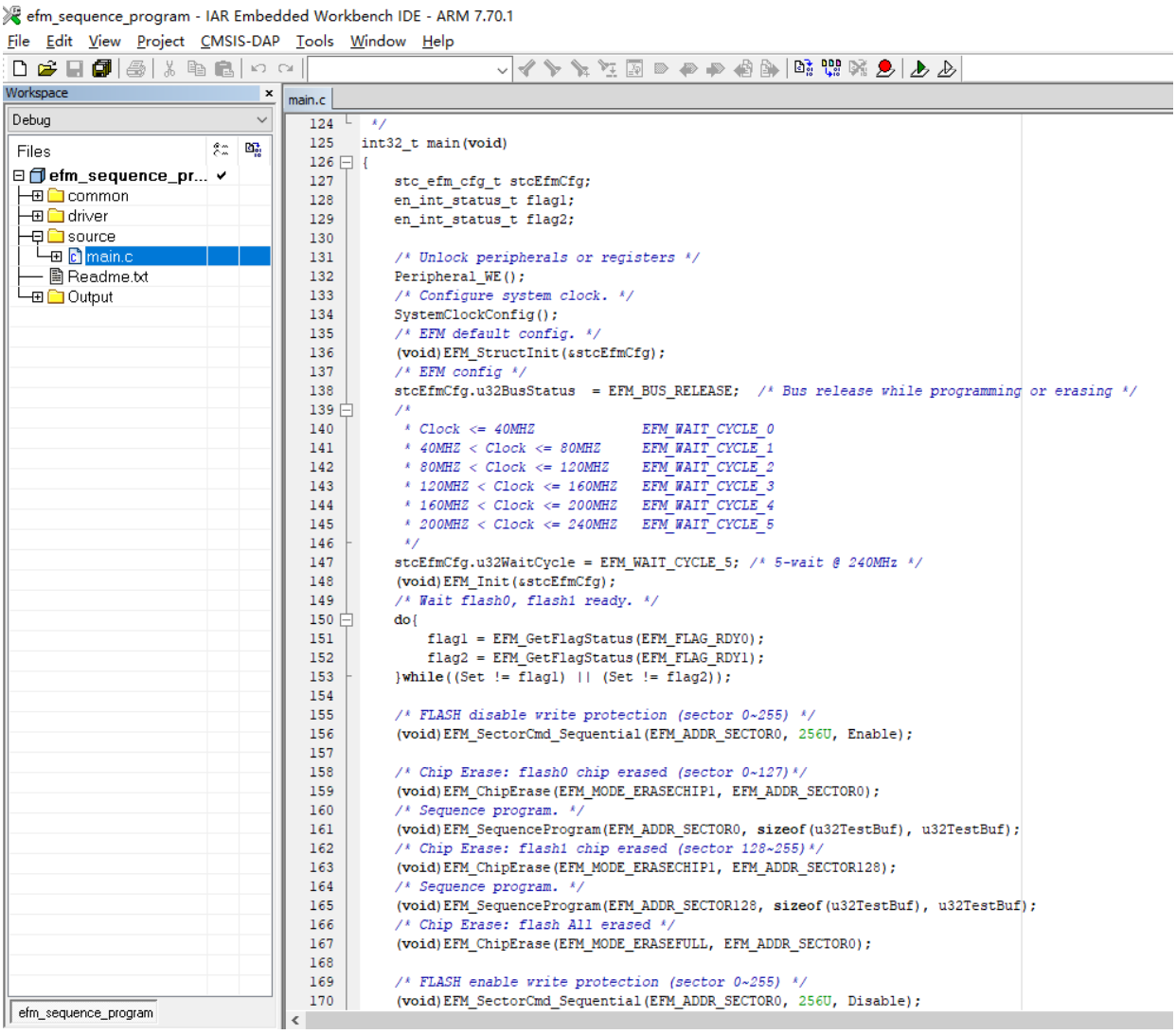

图 4-1 工程视图

2) 修改 IAR 配置，使程序在 RAM 上运行，具体操作步骤见样例 readme.txt 所述。

3) 点击  重新编译整个项目。

4) 点击  将代码下载到评估板上，分别在 159 行、161 行、163 行、165 行和 167 行代码处打断点，全速运行。

5) 运行程序，在 memory 窗口可分别观察到 flash0 全擦除—>flash0 连续编程—>flash1 全擦除—>flash1 连续编程—>flash0 和 1 全擦除。

5 版本信息 & 联系方式

日期	版本	修改记录
2021/7/27	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

网址：http://www.hdsc.com.cn/mcu.htm

通信地址：上海市浦东新区中科路 1867 号 A 座 10 层

邮编：201203