Shanghai Fudan Microelectronics Group Company Limited

复旦微电子

FM33 车用系列 MCU

应用笔记

应用代码保护功能

V1.0.0

本资料是为了让用户根据用途选择合适的上海复旦微电子集团股份有限公司（以下简称复旦微电子）的产品而提供的参考资料，不转让属于复旦微电子或者第三者所有的知识产权以及其他权利的许可。

在使用本资料所记载的信息最终做出有关信息和产品是否适用的判断前，请您务必将所有信息作为一个整体系统来进行评价。

采购方对于选择与使用本文描述的复旦微电子的产品和服务全权负责，复旦微电子不承担采购方选择与使用本文描述的产品和服务的责任。除非以书面形式明确地认可，复旦微电子的产品不推荐、不授权、不担保用于包括军事、航空、航天、救生及生命维持系统在内的，由于失效或故障可能导致人身伤亡、严重的财产或环境损失的产品或系统中。

今后日常的产品更新会在适当的时候发布，恕不另行通知。 在购买本资料所记载的产品时，请预先向复旦微电子在当地的销售办事处确认最新信息，并请您通过各种方式关注复旦微电子公布的信息，包括复旦微电子的网站(http://www.fmsh.com/)。

如果您需要了解有关本资料所记载的信息或产品的详情，请与上海复旦微电子集团股份有限公司在当地的销售办事处联系。

商 标

目　　录

1 说明

FM33 车用系列 MCU 设计有应用代码保护功能，本应用笔记主要描述用户在开发该功能时所需注意的事项。

2 功能简介

2.1 读写保护

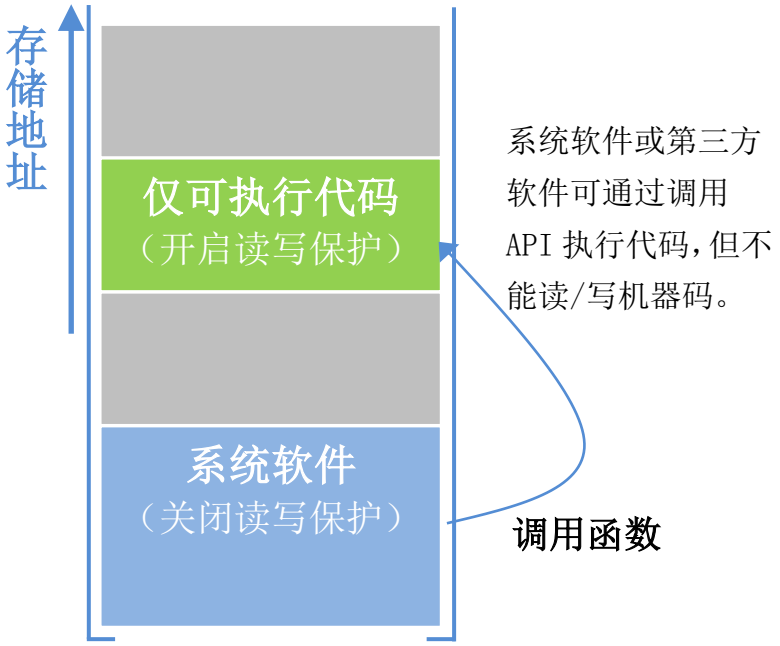
FM33 车用系列 MCU 的 Code Flash 以 8k 字节为一个存储单元（Block），支持对每个存储单元开启读/写保护。

表一 应用代码读写保护

类型 / LOCK(2bit)	SWD 全擦	SWD 扇/页擦	SWD 写	SWD 读	软件 扇/页擦	软件 写	软件 读
11：无保护	√	√	√	√	√	√	√
01/10：软件读写保护，仅取指	√	√	√	√	×	×	×
00：软件读写保护，仅取指；SWD 保护	√	×	×	×	×	×	×

2.2 典型应用

用户可以将需要保护的代码先独立封装（提供 API），然后存储在指定单元并使能读/写保护，此时存储保护单元中的机器码无法被软件或 SWD 口读/写，仅支持以 API 的方式被调用执行。



图一 典型应用示意图

该保护机制是允许取指但不允许读写，但嵌入式软件在访问向量表过程中会产生数据访问，以及编译器的部分库函数会存在文字池，因此该保护机制不能用于保护整个软件代码，被保护的软件代码也必须满足相关条件后才能放入使能保护的存储单元。

2.3 软件读处理

使能读/写保护的存储单元禁止读取操作，软件运行时读取方式分为文字池读取和 RO-DATA 读取。

2.3.1 文字池读取

文字池是代码段中的常量数据区域（如某些变量的地址），其属于 RO-CODE 属性，由于没有单条指令可以支持直接生成 32bit 常量，所以编译器生成的代码通过使用伪指令(如 LDR)从文字池中加载这些常量，此方式可以有效减小代码量并提高性能，文字池包含整数型文字池、字符串文字池、浮点数文字池以及分支表。

如存储保护单元中存在文字池，则 CPU 因无法从这部分文字池中加载正确数据而导致程序运行异常。此应对策略是被保护代码禁止生成文字池，具体请见开发注意事项。

2.3.2 RO-DATA 读取

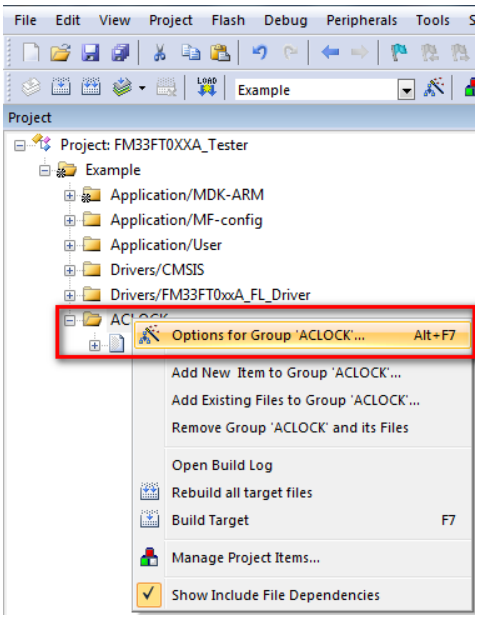
用户应用代码定义的常量属于 RO-DATA 属性，代码通过数据加载方式访问，编译器不能改变其属性和访问方式。

如存储保护单元中存在 RO-DATA，则 CPU 加载这部分 RO-DATA 会得到一个错误的数值（固定为 0x55555555），此不会影响程序执行，但系统软件的表现行为由软件的鲁棒性决定。比如定义了一个整型变量的指针常量，由于读/写保护机制加载时得到 0x55555555 数值，如软件无冗余处理措施而直接指针访问则会因地址非对齐访问而触发 HardFault，此应对策略是被保护代码中的 RO-DATA 指定存储地址且不能放入存储保护单元，具体请见开发注意事项。

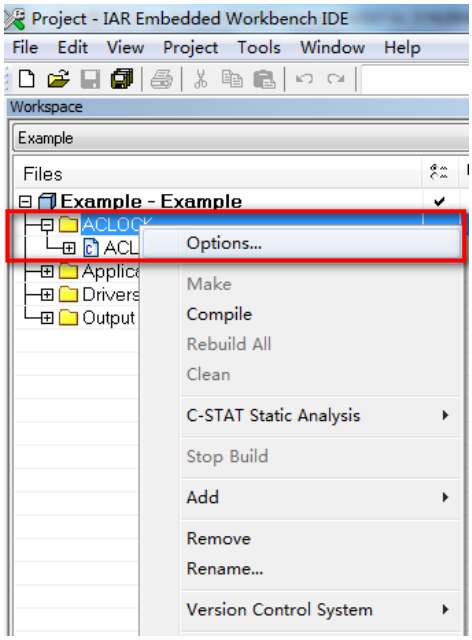
3 开发注意事项

3.1 创建独立文件组和.c 文件

建议被保护代码创建独立的文件组和.c 文件，此举的目的是方便对被保护代码设置私有属性，而非对整个工程设置。



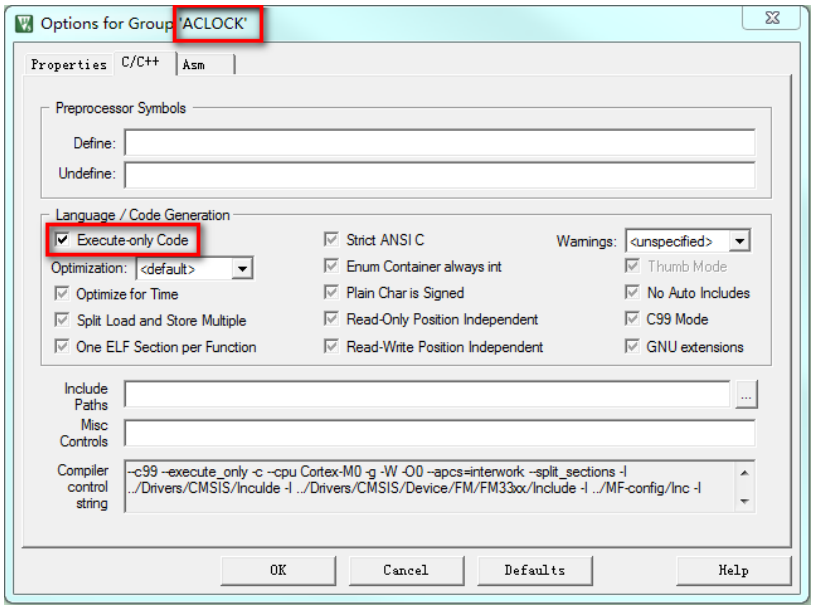
图二 Keil 文件组私有属性设置示意



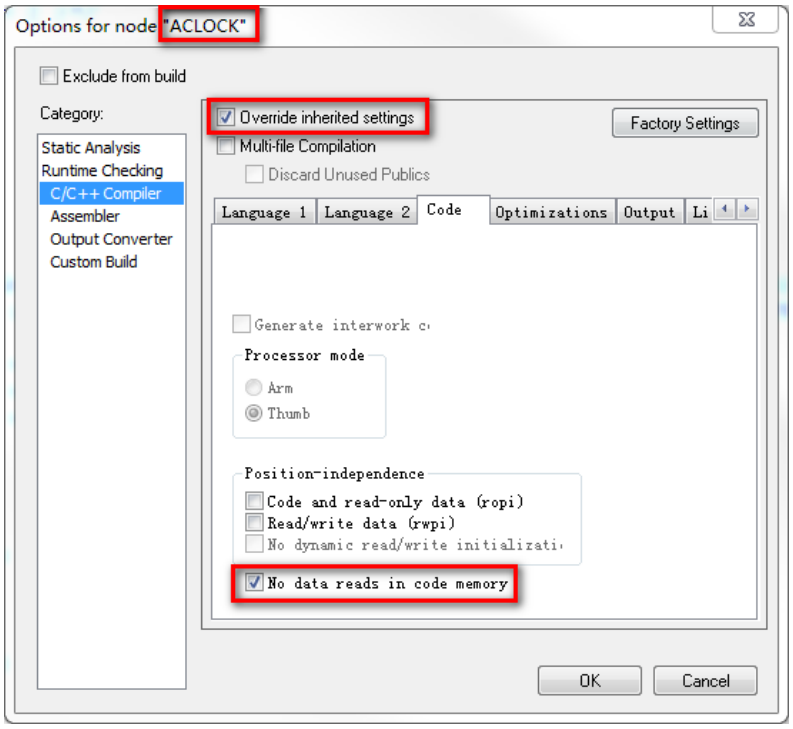
图三 IAR 文件组私有属性设置示意

3.2 禁止生成文字池

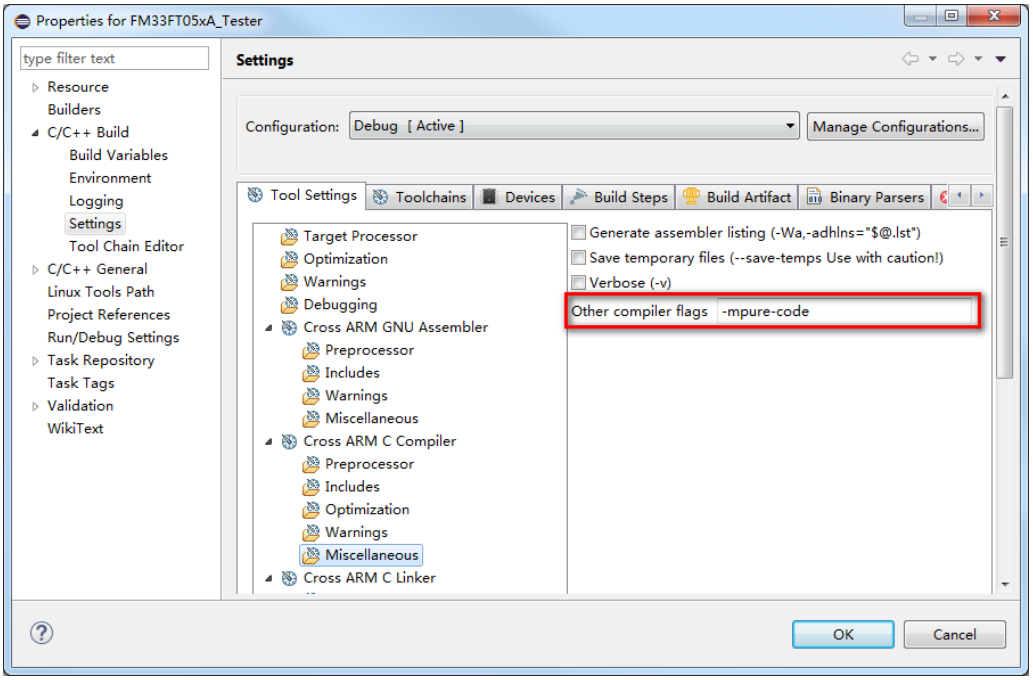
每个工具链都有其自己的选项，来防止编译器生成文字池和分支表，配置如下：



图四 Keil 编译器对单个文件组设置私有属性



图五 IAR 编译器对单个文件组设置私有属性



图六 GCC 编译器配置界面

----------**注意**----------

在禁止生成文字池后 GCC 编译器为了生成代码量不显著增加，其使用 MOVW 和 MOVT 指令生成 32bit 常量，由于 Armv6-M 架构不支持 MOVW 和 MOVT 指令，因此 GCC 编译器不适合 Cortex-M0 和 Cortex-M0+ 处理器关于这方面的处理。

----------**注意**----------

该应用笔记不包含对汇编源文件的处理建议。

代码量增加：

当禁止生成文字池后编译器为了生成代码量不显著增加，针对 Armv7-M 或 Armv8-M 架构，编译器会优先使用 MOVW 和 MOVT 32bit 指令(指令中包含 16 位数据)生成 32bit 常量。

Armv6-M 架构不支持 MOVW 和 MOVT 32bit 指令，编译器的典型策略是使用 LSLS 和 ADDS 指令对多个单字节立即数进行多次左移和相加最终生成 32bit 常量，其所带来的影响是代码量显著增加和运行效率降低。Cortex-M0 和 Cortex-M0+ 处理器属于 Armv6-M 架构，FM33 系列 MCU 使用的是 Cortex-M0+处理器，因此用户需要注意到这方面的影响。

```
   924:          FLASH->ACLOCK1 = ACLOCK[0];
0x00002C72 480A       LDR      r0,[pc,#40]  ; @0x00002C9C
0x00002C74 6800       LDR      r0,[r0,#0x00]
0x00002C76 490A       LDR      r1,[pc,#40]  ; @0x00002CA0
0x00002C78 6208       STR      r0,[r1,#0x20]
```

地址0x00002C9C区域是文字池，其存放的是一个常量的地址，先获取地址，再通过地址加载常量到CPU寄存器

图七 使用文字池的汇编代码

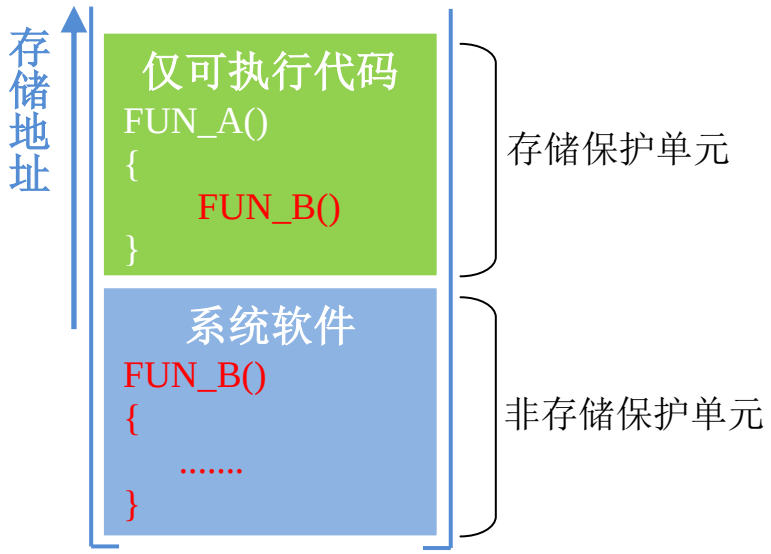
```
   924:          FLASH->ACLOCK1 = ACLOCK[0];
0x0003D396 2000       MOVS     r0,#0x00
0x0003D398 0200       LSLS     r0,r0,#8
0x0003D39A 3000       ADDS     r0,r0,#0x00
0x0003D39C 0200       LSLS     r0,r0,#8
0x0003D39E 3024       ADDS     r0,r0,#0x24
0x0003D3A0 0200       LSLS     r0,r0,#8
0x0003D3A2 30D0       ADDS     r0,r0,#0xD0
0x0003D3A4 6800       LDR      r0,[r0,#0x00]
0x0003D3A6 2140       MOVS     r1,#0x40
0x0003D3A8 0409       LSLS     r1,r1,#16
0x0003D3AA 3110       ADDS     r1,r1,#0x10
0x0003D3AC 0209       LSLS     r1,r1,#8
0x0003D3AE 3140       ADDS     r1,r1,#0x40
0x0003D3B0 6208       STR      r0,[r1,#0x20]
```

禁止文字池后，常量的地址被编译成多个单字节立即数。
加载常量时先将多个单字节立即数组合得到该常量的地址，再通过地址加载常量到CPU寄存器。

图八 禁止文字池的汇编代码

3.3 函数调用

存储保护单元的代码也可以调用非存储保护单元中的函数，但由于存储保护单元的代码禁止读/写，所以其跳转的目标函数指针值固定不可改变，因此系统软件要确保非存储保护单元中被调用的函数地址不能改变，否则程序将工作异常。



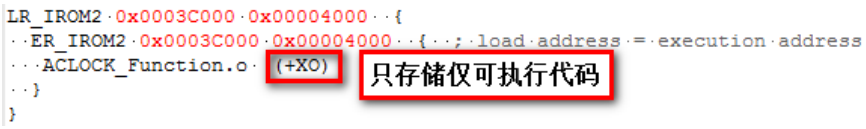
图九 存储保护单元代码调用非存储保护单元的函数

3.4 库函数

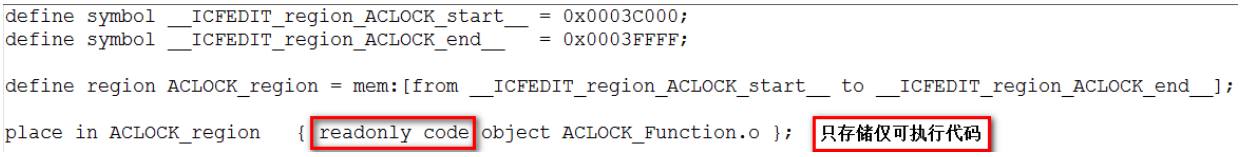
即使编译器禁止生成文字池，但编译器的部分库函数生成的机器码可能仍不符合仅可执行代码的标准，如被保护代码调用了此类库函数且定义到存储保护单元中，则程序将不会正常工作。因此如使用了库函数则应检查是否存在文字池。

3.5 分配存储单元及定义存储属性

被保护代码必须通过编译器的 MEMORY 文件分配到指定存储区域，然后通过编程器使能该存储区域的读/写保护功能（存储保护单元）。MEMORY 文件分配存储区域时还必须定义其存储属性为仅可执行代码。



图十 Keil 编译器.sct 文件编辑



图十一 IAR 编译器.icf 文件编辑

3.5.1 RO-DATA 存储

被保护代码中如定义有常量（如整数、浮点数和字符串），则常量不能放到存储保护单元，通过上面 MEMORY 文件对存储属性（仅可执行代码）定义，RO-DATA 不会存储到指定区域（存储保护单元），其实际存储地址由编译器自动分配。

以上配置可以保证被保护代码中的常量不会存储到保护单元，确保了程序正常运行。但由于存储保护单元的代码是固定的，因此其加载常量的目标地址也是固定的，如果非存储保护单元中的该常量地址被改变将导致数据加载错误，此不会影响程序执行，但系统软件地表现行为由软件的鲁棒性决定。

基于软件鲁棒性考虑，我们建议被保护代码中的常量统一固定地址并分配的非存储保护单元中。关于常量地址的指定方法较多,可以通过 MEMORY 文件指定，也可通过关键字指定，示例：

Keil 将常量指定在固定地址：

```
/* 将常量 k 放在 Flash 的 0x00002000,该存储单元禁止保护 */
const int k __attribute__((at(0x00002000))) = 255;
```

IAR 将常量指定在固定地址：

```
/* 将常量 k 放在 Flash 的 0x00002000,该存储单元禁止保护 */
const int k @ 0x00002000 = 255;
```

3.6 启动文件及向量表禁止保护

启动文件及向量表总是包含数据加载操作，即使禁止生成文字池也不会改变此属性，编译后启动文件及向量表的代码总是处于机器码的头部，因此机器码的第 1 个 8k（Block0）必须禁止应用代码保护功能。如有在线升级功能则 BOOT 和 APP 的机器码均按此要求执行。

3.7 FLASH KEY 错误锁定状

如系统软件对存储保护单元执行擦除/编程操作，则 Flash 会处于 KEY 错误锁定状态(FLASH->ISR：KEYSTA)，整片 Flash 不再支持擦除/编程操作，必须执行复位才能恢复。因此如系统软件有对非存储保护单元执行擦除/编程功能的，必须有判断 KEY 错误锁定状态及复位的冗余措施。

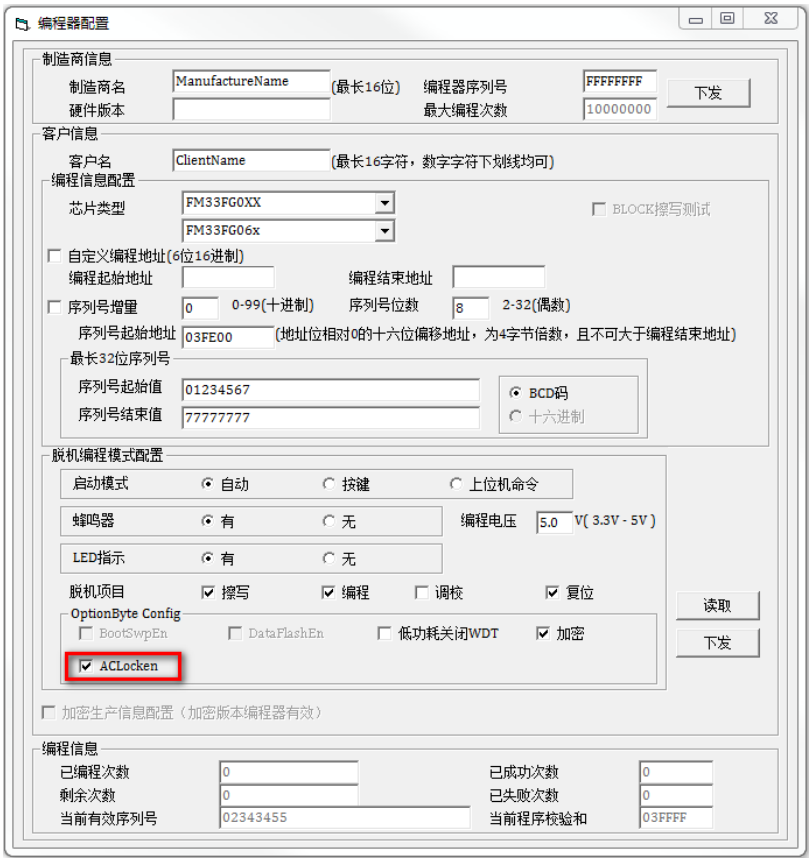
4 编程器配置

为方便软件开发人员调试应用代码，MCU 出厂时默认关闭应用代码保护功能，在受保护的代码确定后由编程器编辑用户配置信息区 LDT1 页开启此功能，此步骤亦可在烧录机器码时同步操作。

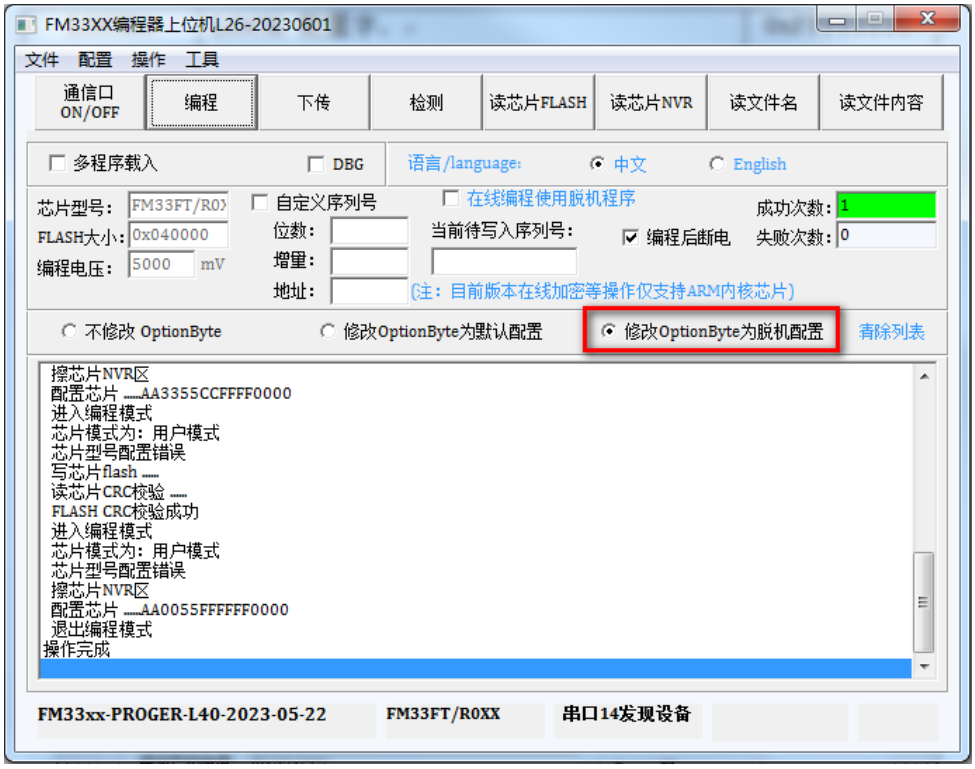
LDT1 页为用户配置信息区，用户通过编程器(SWD)改写，配置信息简介如下：

表二 用户配置信息

配置信息	功能描述	出厂默认
ACLOCKEN	应用代码保护使能（仅针对 code flash） 0x33：禁止 ACLOCK 其他：使能 ACLOCK	0x33
LOCK1	LOCK 配置字， 控制 0~ 127KB flash	0xFFFFFFFF
LOCK2	LOCK 配置字， 控制 128~ 255KB flash（如有）	0xFFFFFFFF
LOCK3	LOCK 配置字， 控制 256~ 383KB flash（如有）	0xFFFFFFFF
LOCK4	LOCK 配置字， 控制 384~ 512KB flash（如有）	0xFFFFFFFF



图十二 FM33XX 编程器上位机使能 ACLOCK

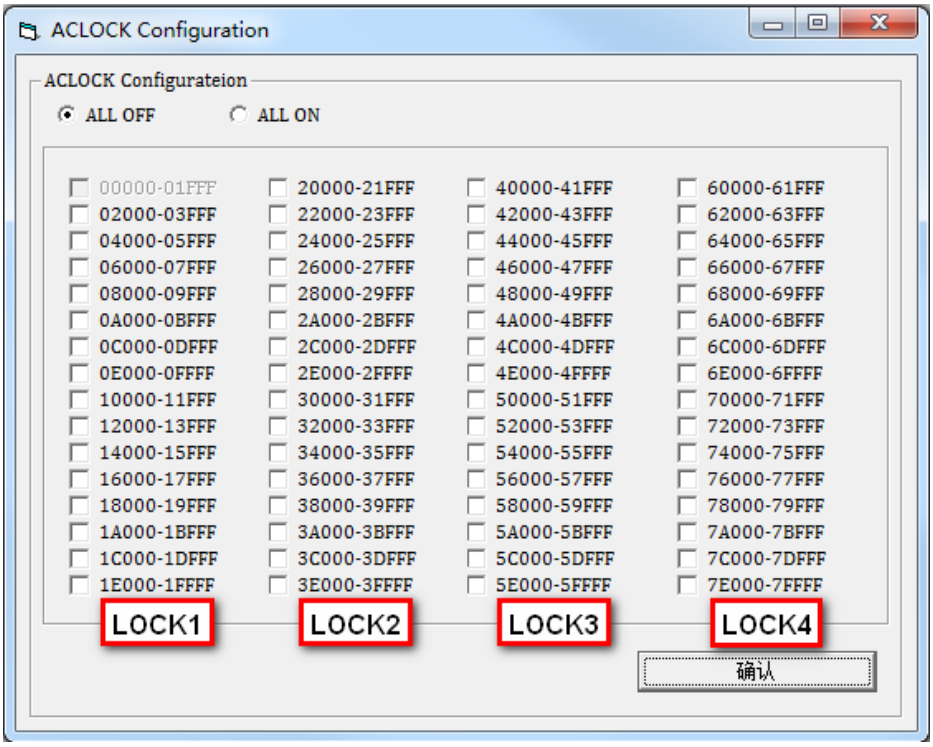


图十三 FM33XX 编程器烧录时同步修改用户配置信息区

一个 LOCK 配置字控制 128KB Flash，每 2bit 对应 8KB（Block）。

表三 LOCK 配置字

位域	功能描述	出厂默认
31:30	11：无保护 01、10：软件读写保护，仅取指 00：软件读写保护，仅取指；SWD 保护 Block15，控制第 120~127kB Flash，共 8KB	b11
29:28	…… Block14，控制第 112~119kB Flash，共 8KB	b11
……	…… ……	b11
3:2	…… Block1，控制第 8~15kB Flash，共 8KB	b11
1:0	…… Block0，控制第 0~7kB Flash，共 8KB	b11



图十四 FM33XX 编程器上位机 LOCK 配置字

关于用户配置信息区更详细信息请参考 MCU 产品说明书中的总线与存储章节。

4.1 LOCK 配置字加载及更改

MCU 每次上电，用户配置信息区的 LOCK 配置字会自动加载到总线与存储外设模块的 FLASH->ACLOCK 寄存器，在程序运行后该寄存器仍可由软件写 0，但不能写 1，因此关于 LOCK 配置字可由编程器编辑也可以由用户软件写 0 使能。

如软件工程有在线升级功能，则用户配置信息区 LOCK 配置字针对 APP 的 Block 不能预先使能应用代码保护功能，而应在 APP 升级成功后由软件通过对 FLASH->ACLOCK 寄存器写 0 使能 APP Block 的应用代码保护功能。

以下 Block 禁止开启应用代码保保护功能：

- 存储启动文件及向量表的 Block，即 BOOT 或 APP 机器码的前 8k 区域；
- 存储 RO-DATA 的 Block，即指定存储 RO-DATA 的 MEMORY 区域；

版本信息

版本号	发布日期	更改说明
1.0.0	2023.08	首次发布

上海复旦微电子集团股份有限公司销售及服务网点

上海复旦微电子集团股份有限公司
地址：上海市国泰路 127 号 4 号楼
邮编：200433
电话：(86-021) 6565 5050
传真：(86-021) 6565 9115

上海复旦微电子（香港）股份有限公司
地址：香港九龙尖沙咀东嘉连威老道 98 号东海商业中心 5 楼 506 室
电话：(852) 2116 3288　2116 3338
传真：(852) 2116 0882

北京办事处
地址：北京市东城区东直门北小街青龙胡同 1 号歌华大厦 B 座 423 室
邮编：100007
电话：（86-10）8418 6608
传真：（86-10）8418 6211

深圳办事处
地址：深圳市华强北路 4002 号圣廷苑酒店世纪楼 1301 室
邮编：518028
电话：(86-0755) 8335 0911　8335 1011　8335 2011 8335 0611
传真：(86-0755) 8335 9011

台湾办事处
地址：台北市 114 内湖区内湖路一段 252 号 12 楼 1225 室
电话：(886-2) 7721 1889
传真：(886-2) 7722 3888

新加坡办事处
地址：237, Alexandra Road, #07-01, The Alexcier, Singapore 159929
电话：(65) 6472 3688
传真：(65) 6472 3669

北美办事处
地址：2490 W. Ray Road Suite#2 Chandler, AZ 85224 USA
电话：(480) 857-6500 ext 18
公司网址：http://www.fmsh.com/