



复旦微电子

FM33LF0xxA ***车用系列MCU***

FLASH 模拟 EEPROM

V1. 3. 1. 3



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。

未经复旦微电子的许可，不得翻印或者复制全部或部分本资料的内容。

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

上海复旦微电子集团股份有限公司的公司名称、徽标以及“复旦”徽标均为上海复旦微电子集团股份有限公司及其分公司在中国的商标或注册商标。

上海复旦微电子集团股份有限公司在中国发布，版权所有。



目 录

1 说明	1
1.1 FLASH 与 EEPROM 之间的主要区别	1
1.1.1 写访问时间的区别	1
1.1.2 写方法的区别	1
1.1.3 擦除时间的区别	2
2 模拟 EEPROM 的实现	3
2.1 原理	3
2.2 应用举例	4
3 软件描述	5
4 示例程序说明	8
4.1 PAGE 基本信息	8
4.2 有效变量的写入过程	8
4.3 可擦写性能	10
4.4 注意事项	11
版本信息	12
上海复旦微电子集团股份有限公司销售及服务中心	13



图 目 录

图 2-1 Page0 和 Page1 之间的页状态切换	3
图 2-2 EEPROM 变量格式化	4
图 2-3 数据更新流程	5
图 3-1 写变量流程图	7
图 4-1 PAGE 配置信息定义	8
图 4-2 PAGE 状态信息定义	8
图 4-3 有效数据个数及地址	9
图 4-4 PAGE0 写满	9
图 4-5 写入第 256 个元素	10
图 4-6 SVD 电压监测示例	10

表 目 录

表 1-1 FLASH 和 EEPROM 之间的主要区别	1
表 4-1 模拟 EEPROM 中储存变量的最大值	11



1 说明

1.1 FLASH 与 EEPROM 之间的主要区别

电可擦编程只读存储器（EEPROM）是很多应用系统的关键组成部分，这些应用系统在进行数据存储时要求一个字或者一个字节地更新数据。而 MCU 在系统中一般运用 FLASH 进行数据保存。为了降低硬件成本，可用 FLASH 代替 EEPROM 进行数据存储。

与 FLASH 不同，EEPROM 在重写数据的时候无需执行一次擦除动作去挪出存储空间。所以用 FLASH 模拟 EEPROM 时需设计软件方案对 FLASH 进行存储空间管理。运用 FLASH 模拟 EEPROM 需考虑很多因素，包括 EEPROM 的可靠性、所使用的 FLASH 的特性以及软件接口需求。

FLASH 与 EEPROM 的主要区别如表 1-1 所示。

表 1-1 FLASH 与 EEPROM 的主要区别

特性	外部 EEPROM	FLASH
写时间	— ms级	字编程时间：25us 左右
擦时间	N/A	页/Sector 擦除时间：2ms
写特性	— 只要开始，便不再依靠CPU	— CPU复位会中止写动作。
读访问	— 100us	— 100ns
写/擦周期	— 10K~1000K Cycle	— 10K~100K Cycle

1.1.1 写时间的区别

相比于EEPROM，FLASH数据写入时间较短，关键数据可以更快地被存储至FLASH模拟的EEPROM中去。

1.1.2 写方法的区别

外部EEPROM 与FLASH模拟EEPROM 在应用上最主要区别就是写方法。



外部EEPROM：一旦启动，写动作不会被CPU 复位中断。只有断电可中断写过程，所以在EEPROM 内，适当大小的退偶电容可以获得完整的写动作过程。

使用FLASH模拟EEPROM：一旦启动，写动作可以被断电或CPU 复位中断。

1.1.3 擦除特性的区别

擦除时间 EEPROM 和使用 FLASH 模拟 EEPROM 之间另一个主要区别。与 FLASH 不同，EEPROM 在写之前无需执行擦除动作去挪出空间。另外，当 FLASH 执行擦除动作时，掉电或其他事件可能中断擦除操作（例如复位）。

2 模拟 EEPROM 的实现

2.1 原理

考虑到 FLASH 的局限性和应用需求，模拟 EEPROM 有多种执行方式。下面所介绍的方案至少需要两页 FLASH。一页为有效 EEPROM 页，支持字节编程；另一页用于有效页数据存满时存储有效数据。

每页占用第一个字段用于表明页状态，每页有三种可能的状态：

ERASED：此页为空。

RECEIVE_DATA：正在从另一满页接受数据。

VALID_PAGE：此页包含有效数据，且此状态不会改变，直到所有的有效数据全部被转移到另一页。

页与页之间的状态切换如图2-1所示。

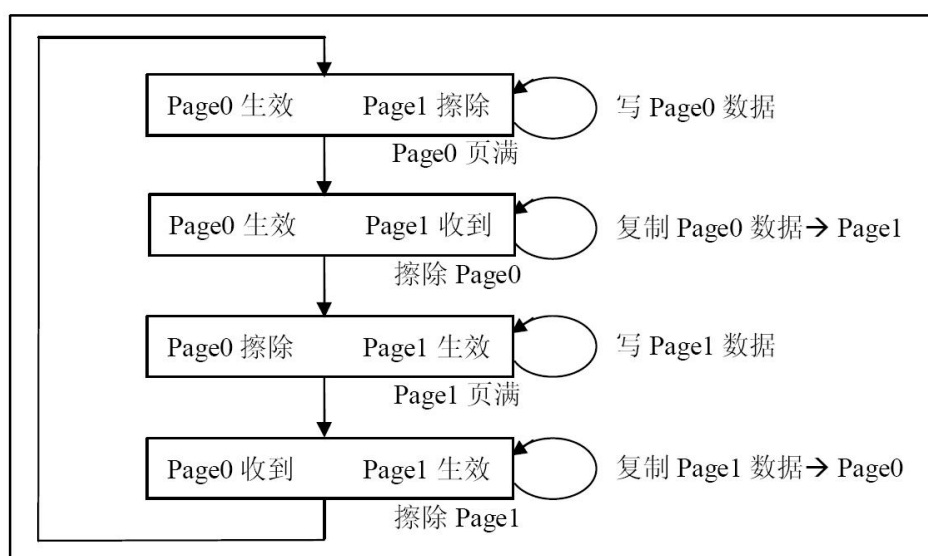


图 2-1Page0 和 Page1 之间的页状态切换

为方便后续软件读写数据，每个变量元素包含一个虚拟地址和一个存储在 EEPROM 中的值（在编译器软件中，虚拟地址和数据都是 16 位长）。当同一虚拟地址的数据被

修改时，此次虚拟地址和数据被存储至一个新的 FLASH 地址。读取数据时，仅检索最后一个虚拟地址上的数据。

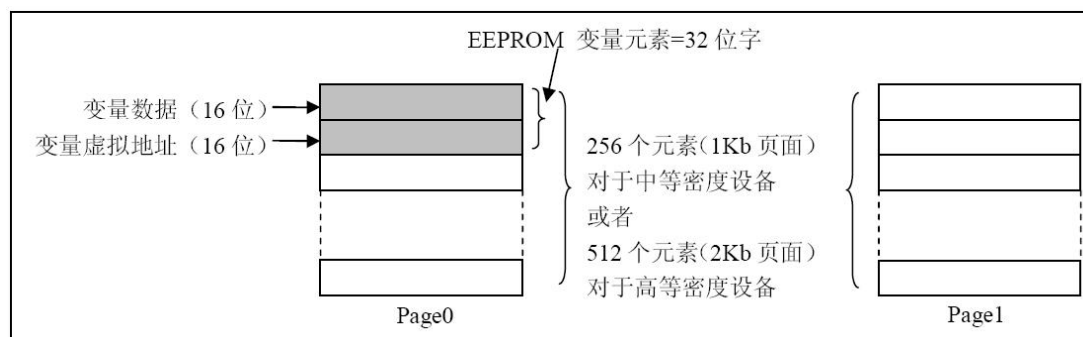


图 2-2 EEPROM 变量格式化

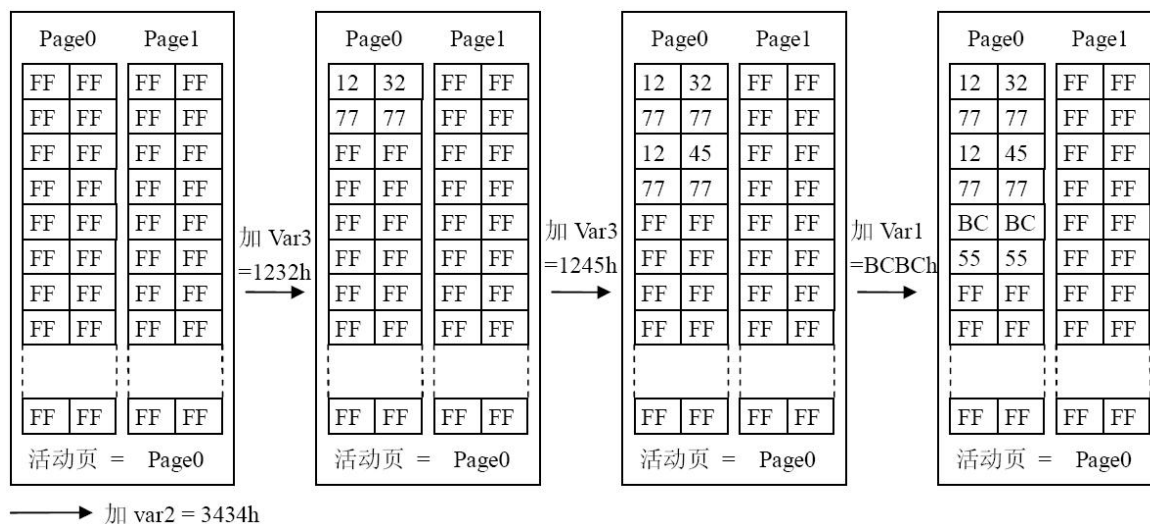
2.2 应用举例

下面的例子展示了三个EEPROM 变量 (Var1, Var2 和Var3) 与虚拟地址之间的管理方案:

Var1: 5555h

Var2: 6666h

Var3: 7777h



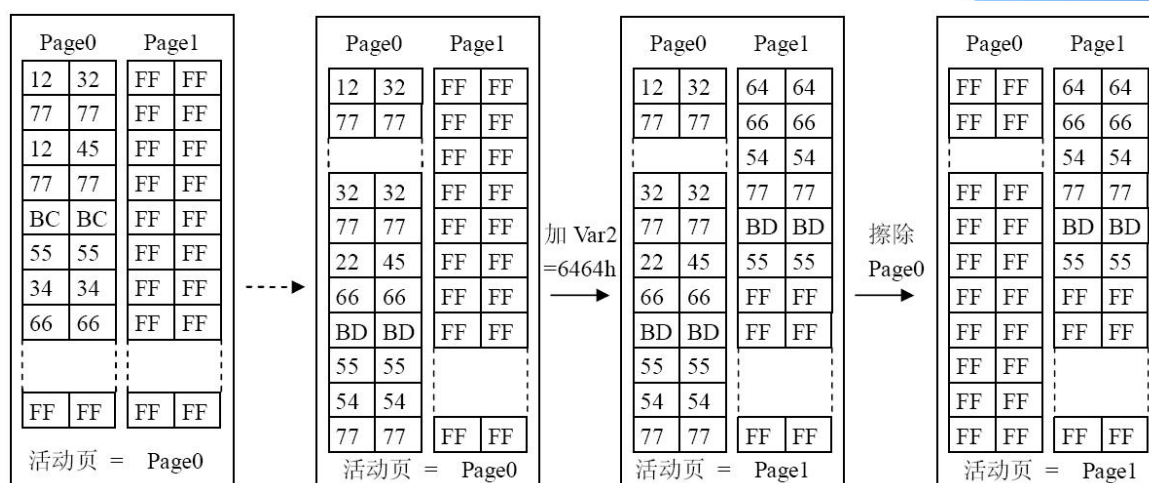


图 2-3 数据更新流程

3 软件描述

这里使用的三个变量Var1，Var2和Var3 在数组VirtAddVarTab 中进行定义，VirtAddVarTab 数组在程序 main.c 文件中进行声明。

除了闪存的库文件，这个工程包含了三个源文件：

- **eeeprom.c:** 包含以下函数原型：

```
EE_Init()
EE_Format()
EE_FindValidPage()
EE_VerifyPageFullWriteVariable()
EE_ReadVariable()
EE_PageTransfer()
EE_WriteVariable()
```

- **eeeprom.h:** 包含一些宏定义及函数声明。

- **main.c:** 用于展示FLASH模拟EEPROM的读写接口的使用方法。

用户API 定义函数的设置在eeeprom.c 文件内，然后用于模拟EEPROM，描述如下：

- **EE_Init()**

扇区擦除或转移数据时候发生掉电，可能会造成扇区起始标志位发生错误。在这种情况下，EE_Init()函数将会尝试将FLASH数据初始化至正常运行状态。每一次断电后访问数据之前，这个函数都会被调用。

● EE_Format()

此函数擦除page0 和page1 并且写有效页标志位至page0。

● EE_FindValidPage()

此函数读取所有页的数据标志并返回有效页页码。这个返回的参数表明了有效页是否在执行写入或读取操作(READ_FROM_VALID_PAGE or WRITE_IN_VALID_PAGE)。

● EE_VerifyPageFullWriteVariable()

此函数为从最后一位开始找到有效页的第一个空位置，并写入虚拟地址和变量的数据。在活动页满的情况下，返回页满标志。这段程序用到了以下参数：

- 虚拟地址：可能是任何一个已经声明的虚拟地址（Var1，Var2 或者Var3）
- 数据：将要存储至虚拟地址中的值

这个函数操作成功的话返回操作完成标志，如果没有足够的内存区更新变量，则返回页满，或者返回操作错误标志，用来指示操作失败（擦除或编程）。

● EE_ReadVariable()

此函数返回的是虚拟地址所对应的数据，虚拟地址做为函数形参传入。只有最后更新至EEPROM的有效地址上的数据被读取。若未发现有效地址，读状态变量将返回1。

● EE_PageTransfer()

此函数在当前页满后转移有效地址所对应的元素至另一空白页。当有效数据转移完成后，新页的页标志更改为VALID_PAGE，旧页将被擦除并且它的页标志会变成ERASED。

● EE_WriteVariable(..)

当写入数据至EEPROM中时，此函数将会被调用。它使用EE_VerifyPageFullWriteVariable() 和EE_PageTransfer() 函数。

图4 展示了EEPROM 更新数据变量写入的过程。

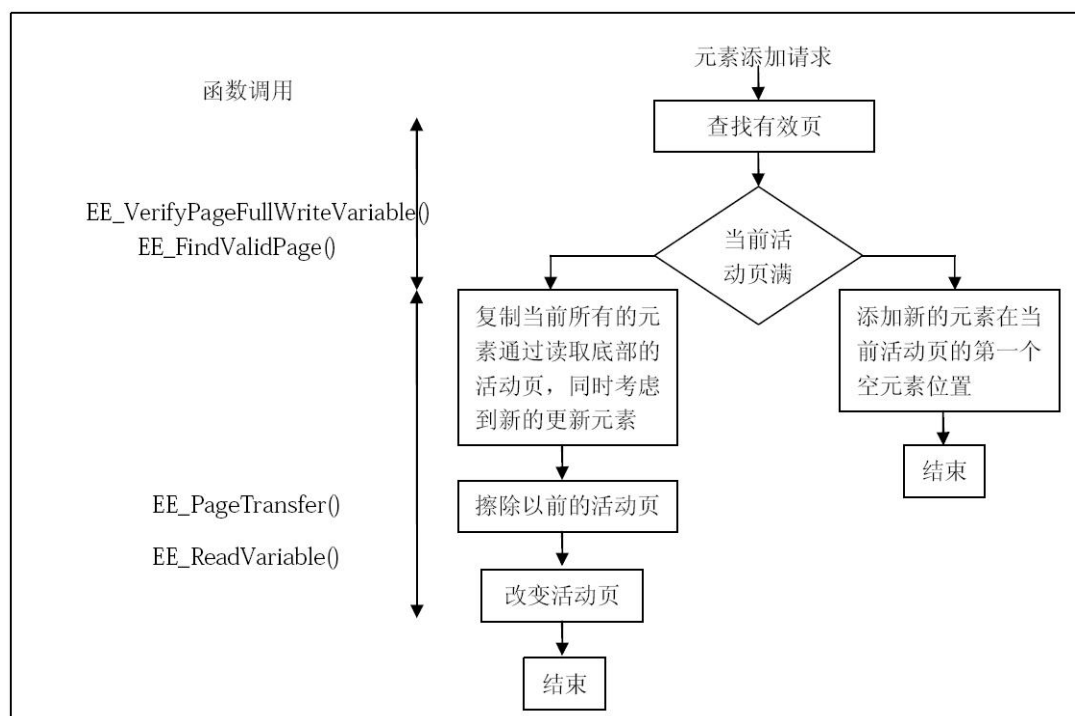


图 3-1 写变量流程图

4 示例程序说明

4.1 PAGE 基本信息

本示例程序使用 PAGE0 与 PAGE1 两页来进行数据交换。用于模拟 EEPROM 的 Flash 起始地址为 0x00010000，PAGE 大小为 2Kbyte。以上 PAGE0、PAGE1 的基本信息均存储于 eeprom.h 文件中。

```
#define PAGE_SIZE ((uint16_t)0x800) /* Page size = 2KByte */

/* EEPROM start address in Flash */
#define EEPROM_START_ADDRESS ((uint32_t)0x00010000) /* EEPROM emulation start address: after 64KByte of used Flash memory */

/* Pages 0 and 1 base and end addresses */
#define PAGE0_BASE_ADDRESS ((uint32_t)(EEPROM_START_ADDRESS + 0x000))
#define PAGE0_END_ADDRESS ((uint32_t)(EEPROM_START_ADDRESS + (PAGE_SIZE - 1)))

#define PAGE1_BASE_ADDRESS ((uint32_t)(EEPROM_START_ADDRESS + PAGE_SIZE))
#define PAGE1_END_ADDRESS ((uint32_t)(EEPROM_START_ADDRESS + (2 * PAGE_SIZE - 1)))
```

图 4-1 PAGE 配置信息定义

PAGE0 与 PAGE1 的页状态可通过 Page status 来判断，页状态存储于每个 PAGE 前 32 个位。程序在执行过程中会读取每页状态。

```
/* Page status definitions */
#define ERASED ((uint16_t)0xFFFF) /* PAGE is empty */
#define RECEIVE_DATA ((uint16_t)0xEEEE) /* PAGE is marked to receive data */
#define VALID_PAGE ((uint16_t)0x0000) /* PAGE containing valid data */
```

图 4-2 PAGE 状态信息定义

4.2 有效变量的写入过程

变量的虚拟地址以及数据均采用 32 位数据写入的编程方式。本例程中的有效变量个数及虚拟地址分别定义于文件 eeprom.h 及 main.c 文件中。其中有效变量个数 NumOfVar 为 10 个，虚拟地址分别为：0x1111, 0x2222, 0x3333, 0x4444, 0x5555, 0x6666, 0x7777, 0x8888, 0x9999, 0xAAAA。

```

/* Variables' number */
#define NumbOfVar ((uint8_t)0x0A)

/* Virtual address defined by the user: 0xFFFF value is prohibited */
uint16_t VirtAddVarTab[NumbOfVar] = {0x1111, 0x2222, 0x3333, 0x4444, 0x5555, 0x6666, 0x7777, 0x8888, 0x9999, 0xAAAA};

```

图 4-3 有效数据个数及地址

例程中，PAGE0 起始地址为 0x00010000，PAGE1 起始地址为 0x00010800。先向 PAGE0 中写入 255 个变量，将 PAGE0 填满。如图 4-4 所示，此时以第 255 个变量为例：0x34 为虚拟地址，0x12 为数据。由于 0x34 没有被定义为有效变量地址，在进行页转换时第 255 个变量会被擦除，而最后虚拟地址为 0x1111、0x2222...x0AAAA 的变量会被转移至 PAGE1。

```

/*----Store successively many values of the three variables in the EEPROM----*/
/*Store 254 values of Variable1 in EEPROM*/
for (VarValue = 0; VarValue < 254; VarValue++)
{
    if ((VarValue % 254) == 0)
    {
        LED0_TOG();
    }
    EE_WriteVariable(VirtAddVarTab[VarValue % NumbOfVar], VarValue);
}

/*Store the 255th value of Variable1 in EEPROM*/
EE_WriteVariable(0x34, 0x12);

/*Store the 256th value of Variable1 in EEPROM*/

```

Memory 1									
Address: 0x00010000									
0x00010700:	000000DF	00004444	000000E0	00005555	000000E1	00006666	000000E2	00007777	
0x00010720:	000000E3	00008888	000000E4	00009999	000000E5	0000AAAA	000000E6	00001111	
0x00010740:	000000E7	00002222	000000E8	00003333	000000E9	00004444	000000EA	00005555	
0x00010760:	000000EB	00006666	000000EC	00007777	000000ED	00008888	000000EE	00009999	
0x00010780:	000000EF	0000AAAA	000000F0	00001111	000000F1	00002222	000000F2	00003333	
0x000107A0:	000000F3	00004444	000000F4	00005555	000000F5	00006666	000000F6	00007777	
0x000107C0:	000000F7	00008888	000000F8	00009999	000000F9	0000AAAA	000000FA	00001111	
0x000107E0:	000000FB	00002222	000000FC	00003333	000000FD	00004444	00000012	00000034	
0x00010800:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	
0x00010820:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	
0x00010840:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	
0x00010860:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	
0x00010880:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	

图 4-4 PAGE0 写满

当写入第 256 个变量时，由于 PAGE0 已满，第 256 个变量被写入 PAGE1 中，PAGE0 中定义的有效变量将被转移至 PAGE1 中（从页尾根据有效地址逐一寻找），随后 PAGE0 被擦除。整个数据的变化过程如图 4-5 所示。

```

/*----Store successively many values of the three variables in the EEPROM----*/
/*Store 254 values of Variable1 in EEPROM*/
for (VarValue = 0; VarValue < 254; VarValue++)
{
    if ((VarValue % 254) == 0)
    {
        LED0_TOG();
    }
    EE_WriteVariable(VirtAddVarTab[VarValue % NumbOfVar], VarValue);
}

/*Store the 255th value of Variable1 in EEPROM*/
EE_WriteVariable(0x34, 0x12);

/*Store the 256th value of Variable1 in EEPROM*/
EE_WriteVariable(0x78, 0x56);

```

Memory 1									
Address: 0x00010000									
0x000107C0:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF
0x000107E0:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF
0x00010800:	00000000	FFFFFFFF	00000056	00000078	000000FA	00001111	000000FB	00002222	
0x00010820:	000000FC	00003333	000000FD	00004444	000000F4	00005555	000000F5	00006666	
0x00010840:	000000F6	00007777	000000F7	00008888	000000F8	00009999	000000F9	0000AAAA	
0x00010860:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	
0x00010880:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	
0x000108A0:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	
0x000108C0:	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	

图 4-5 写入第 256 个元素

4.3 可擦写性能

当使用 EEPROM 的时候，每一个字节可以在一个有限的次数内被编程或擦除，典型范围是 10000 到 100000，FLASH 中最小的擦除尺寸是一页。模拟 EEPROM 的寿命被最经常写的参数的更新速率所限制。循环的能力与用户想要处理的数据的数量和大小有关。在这个例子中，FLASH 用 32 位的数据编程。每一个变量都对应一个 32 位虚拟地址。也就是说，每一个变量占用 2 个字的存储空间。2Kb（对于高密度器件）乘以闪

存可以忍耐的周期数 10000，模拟闪存一个页面的生命周期内 20000Kb 的数据的能力。因此，在仿真过程中，模拟 EEPROM 提供的两个页面可以存储 40000Kb 的数据。如果超过两个页面被使用，这个数量也会相应的增加。当知道所要储存变量数据的宽度的时候，就可以计算出在模拟 EEPROM 的整个生命周期中能够存储的变量的数量。

通过变量的虚拟地址和数据大小，表 3 给出了一个关于模拟 EEPROM 可以存储变量的数量。

变量大小	2 x 1Kb	2 x 2Kb
8 位变量 (8 位虚拟地址)	$10000 \times (2^{10} - 2)$	$10\ 000 \times (2^{11} - 2)$
16 位变量 (16 位虚拟地址)	$5000 \times (2^{10} - 4)$	$5000 \times (2^{11} - 4)$
32 位变量 (32 位虚拟地址)	$2500 \times (2^{10} - 8)$	$2500 \times (2^{11} - 8)$

表 4-1 模拟 EEPROM 中储存变量的最大值

4.4 注意事项

FLASH 操作过程中 VDD 禁止出现低于工作电压或发生复位，否则无法保证 FLASH 中的数据。在进行项目开发时，写操作前请检测当前 VDD 电压，防止低电压下操作。

```
uint16_t EE_WriteVariable(uint16_t VirtAddress, uint16_t Data)
{
    uint16_t Status = 0;

    /* 检测到低压后禁止写操作 */
    if (true == SVD_Result_Confirmed(SVD_BELOW_THRESHOLD, 1U))
    {
        return Status;
    }

    /* Write the variable virtual address and value in the EEPROM */
    Status = EE_VerifyPageFullWriteVariable(VirtAddress, Data);

    /* In case the EEPROM active page is full */
    if (Status == PAGE_FULL)
    {
        /* Perform Page transfer */
        Status = EE_PageTransfer(VirtAddress, Data);
    }

    /* Return last operation status */
    return Status;
}
```

图 4-6 SVD 电压监测示例



版本信息

版本号	发布日期	更改说明
1.3.1.3	2024.12	首次发布



上海复旦微电子集团股份有限公司销售及服务中心

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 Alexcior, 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.fmsk.com/>