Application Note

EEPROM 仿真 B 型设计

Zhao Yuhao

摘要

本应用手册介绍了 B 型 EEPROM 仿真库。本文档描述了 B 型 EEPROM 仿真的结构和行为。然后，介绍了相关 API 的使用。此外，还向用户提供了相关代码。用户可以调用应用程序中的函数来写入、读取和修改数据。

内容

插图清单

表格清单

商标

English Document: SLAAEB4

1 引言

许多应用都需要将数据存储在非易失性存储器中，以便即使在系统再次上电后也可以重复使用或修改应用。EEPROM 是专为此类应用程序设计的。尽管 MSPM0 MCU 没有内部 EEPROM，但 MSPM0 内部闪存支持 EEPROM 仿真。与使用外部串行 EEPROM 相比，使用内部闪存的 EEPROM 仿真可节省引脚用量和成本。

不同应用所需的存储结构不尽相同。本文中介绍的 B 型设计专为存储小*变量*数据而设计。如果应用需要存储大量数据*块*，则可以参考 A 型设计。

1.1 EEPROM 与片上闪存的区别

EEPROM 可以多次擦除和写入存储器的单个字节，即使系统断电，已编程的位置也能长时间保留数据。

闪存的密度高于 EEPROM，因此可以在芯片上实现更大的存储器阵列（扇区）。闪存擦除/写入周期是通过对各个存储单元施加时控电压来执行的。在擦除时，每个单元（位）读取逻辑值 1。因此，每个闪存位置在擦除后的读数为 0xFFFF。通过编程可以将存储单元更改为逻辑 0。可以重写任何字，将位从逻辑 1 更改为逻辑 0；但反过来则不行。此外，闪存有一项限制是必须按区域擦除存储器。对于 MSPM0 MCU，擦除分辨率是大小为 1k 字节的*扇区*。

EEPROM 与闪存的一个主要区别是耐写次数。闪存耐写次数通常为 10000 次，远低于实际 EEPROM。EEPROM 仿真是一种基于闪存的软件，旨在提供可满足应用需求的等效耐写次数。闪存还会仿真 EEPROM 的行为，从而简化读写数据的操作。

典型的仿真方案涉及使用闪存的一部分并将其划分成多个区域。这些区域交替用于存储数据。由于闪存需要按块进行擦除，因此必须保留完整的闪存扇区用于 EEPROM 仿真。为了实现磨损均衡，至少使用两个扇区。

2 实现

2.1 原理

在本文的解决方案中，闪存被划分为多个称为*数据项*的区域。每个数据项为 8 字节，包含数据、写入结束标志和标识符。标识符用于标识和区分数据，类似于变量名或虚拟地址。闪存中的两个数据项可以具有相同的标识符，但只有最新的数据项有效。由于数据项是按顺序写入的，因此可以根据位置区分旧数据项和新数据项。

执行读取操作时，会根据标识符找到相应的数据项。如果有多个具有相同标识符的数据项，则只有最新的数据项有效。执行写入操作时，输入的数据和标识符将汇编为新的数据项。因此，当用户想要更新与标识符相对应的数据项时，写入操作不会修改之前的数据项，而是继续在未使用的区域中创建新的数据项。通常，如果扇区已满，则会将对应于每个标识符的最新数据项转移到下一个扇区，然后擦除该扇区。举个极端的例子，当用户只更新与某个标识符相对应的数据时，扇区已满后，所有数据项的标识符都相同，转移操作只会转移最后一个数据项。

应当注意的是，由于扇区大小是固定的，因此单个扇区可以容纳的数据项数量是有限和固定的。为了存储更多数据项，扇区被分组到*组*中。同一组中的扇区将一起擦除。当一个组已满时，最新的数据项将转移到下一个组，然后将该组擦除。图 2-1 显示了 EEPROM 仿真的结构。

为了标记组的状态，组的前 8 个字节用作标头。组的其余部分（总组大小减去标头的 8 字节大小）用于存储数据项。一个扇区中的数据项数为（扇区大小 x 一个组中的扇区数/数据项大小 - 1）。例如，如果一个组有 1 个扇区，则可以存储 127 个数据项。如果一个组有 2 个扇区，则可以存储 255 个数据项。如果一个组有 3 个扇区，则可以存储 383 个数据项。

应该强调的是，尽管用户可以使用尽可能多的不同标识符（最多等于数据项的数量），但这可能会导致频繁的转移操作和擦除操作，因此可能会增加系统开销。建议的标识符数量是最大数据项数的三分之一到一半。

共有三个用户可配置的参数，可根据应用要求在 eeprom_emulation_type_b.h 中进行配置。这些参数会影响最大数据项数和耐写次数，稍后将对此进行分析。

- 组数：至少 2 个
- 一个组中的扇区数：至少 1 个
- 扇区地址

此外，在数据项的结构中，写入结束标志用于确保数据项的写入完整性。在写入数据和标识符后，此标志设置为 0x0000。

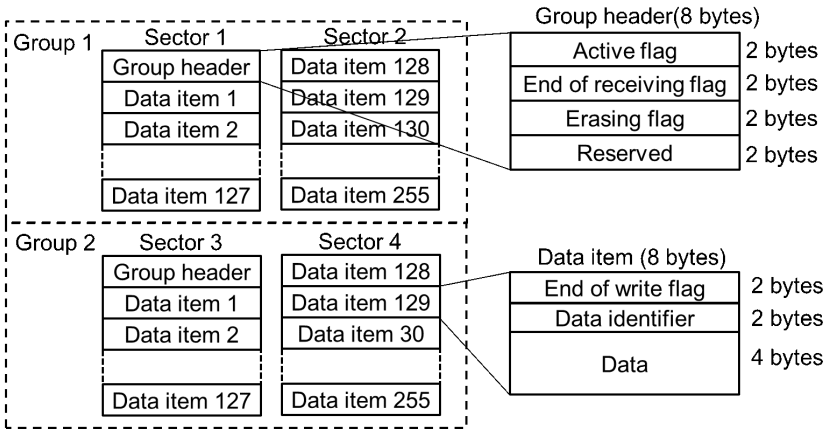

图 2-1. EEPROM 仿真的结构

EEPROM 仿真的基本行为如图 2-2 所示。在 A 中，虽然有两个 Var3，意味着它们具有相同的标识符，但只有后一个标识符有效，因为它较新。如果读取 Var3，则读取的将是 x6 而非 x3。从 A 到 B，执行了写入操作，组 1 已满，因此将执行转移操作。在 C 中，组 2 被标记为 *Receiving*，因此最新的数据项转移到组 2。在 D 中，转移后，组 2 更新为 *Active*。组 1 被标记为“Erasing”并等待擦除。擦除操作仅在用户调用相应函数时执行。

用户可以根据需要选择合适的擦除时间。应该注意的是，不及时擦除将导致试图向有残留数据的扇区写入数据。这可能会导致数据损坏。

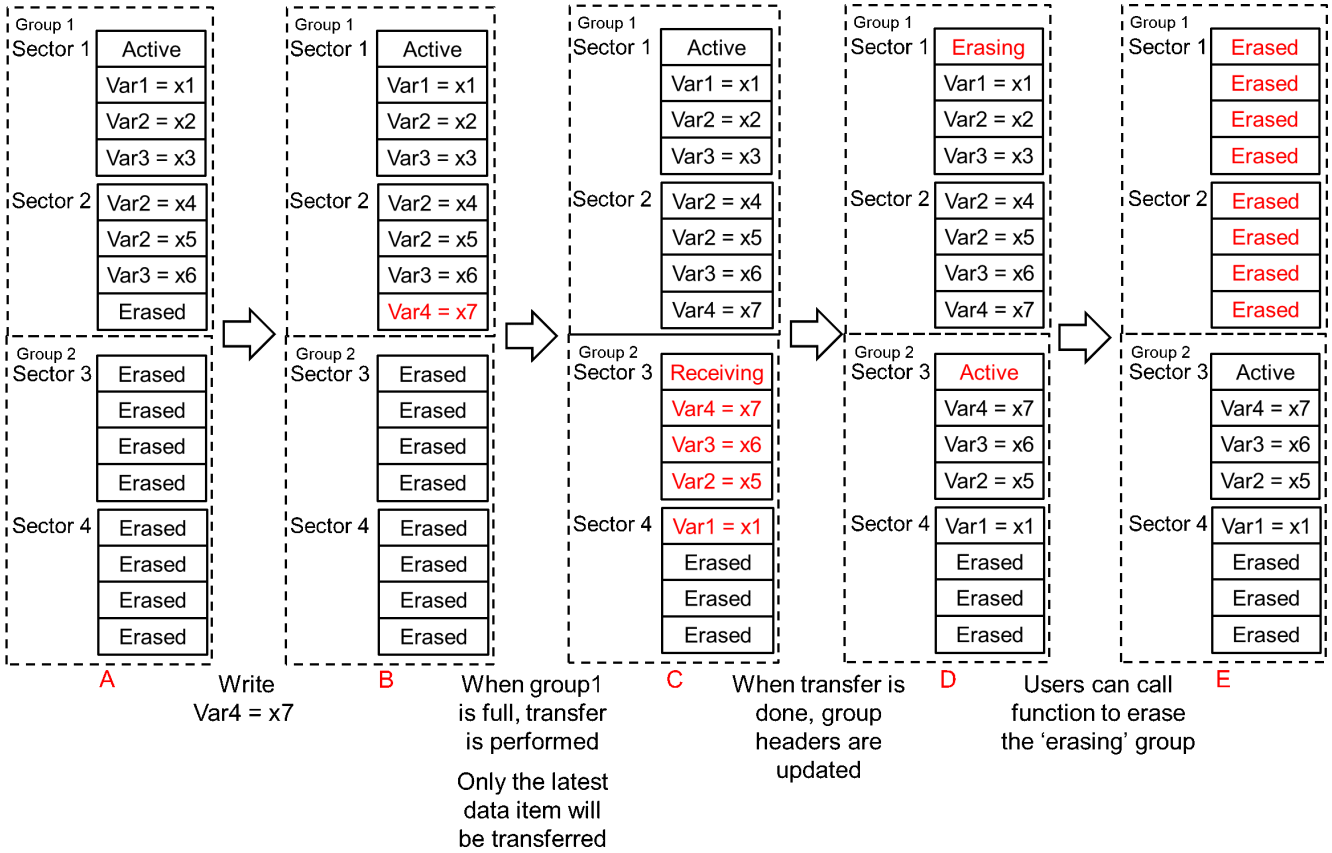

图 2-2. EEPROM 仿真的基本行为

2.2 标头

标头用于组的管理。通过检查单个组的标头，可以确定该组的状态。通过检查所有组的标头，可以找到活动组，并可以检查 EEPROM 仿真的格式。

每个组都有一个标头可显示其状态。标头设置为 8 个字节，具有 3 个标志。根据标志的不同，总共有四种记录状态。标志与记录状态之间的关系如表 2-1 所示。

表 2-1. 标志与记录状态之间的关系

组状态	说明	活动标志	接收结束标志	擦除标志
Erased	擦除后的默认状态	0xFFFF	0xFFFF	0xFFFF
Receiving	转移数据项时，*Receiving* 组接收来自整个组的最新数据项	0x0000	0xFFFF	0xFFFF
Active	*Active* 组是没有填满数据项的组，正在等待写入新的数据项	0x0000	0x0000	0xFFFF
Erasing	*Erasing* 组是等待擦除的组	-	-	0x0000

图 2-3 显示了状态之间如何相互转换。所有标志都会首先被擦除。如果向 *Erased* 组写入数据项，它将更改为 *Active* 状态并等待写入。

如果组已满，则下一个组将更改为 *Receiving* 组，并会将最新的数据项转移到 *Receiving* 组。转移后，整个旧的组将更改为 *Erasing* 状态并等待擦除。然后，*Receiving* 组将变为 *Active* 组。

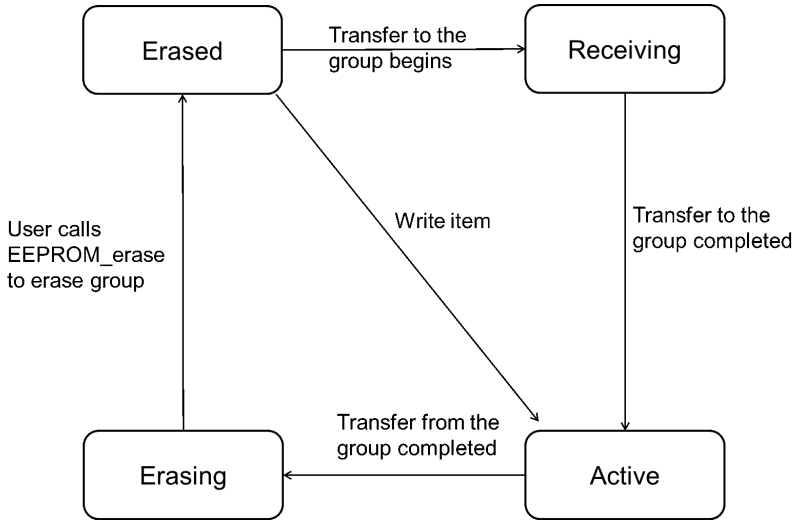

图 2-3. 记录状态的变化

3 软件说明

该软件提供基本的 EEPROM 功能。至少要使用两个扇区来仿真 EEPROM。这些扇区被组织成组，用于存储数据项。四个全局变量用于跟踪 EEPROM 仿真。

3.1 软件功能和流程

总共有六个函数。前四个函数由用户直接调用。最后两个函数由这些函数调用。

- EEPROM_TypeB_init
- EEPROM_TypeB_write
- EEPROM_TypeB_readDataItem
- EEPROM_TypeB_eraseGroup
- EEPROM_TypeB_findDataItem
- EEPROM_TypeB_transferDataItem

图 3-1 中显示了高层软件流程。器件应首先执行初始化代码。通过调用 EEPROM_TypeB_init，它会搜索活动组并检查闪存的格式。如果存在活动组，则会更新全局变量以跟踪活动组和最新数据项。如果活动组不存在，闪存将被初始化。

在应用程序中，用户可以使用 EEPROM_TypeB_readDataItem 根据输入标识符读取数据。用户还可以使用 EEPROM_TypeB_write 来写入数据和标识符。如果该组已满，则最新的数据项将转移到下一个组。转移后，整个组将被标记为 *Erasing* 并设置擦除标志。在下面的流程图中，设置擦除标志后会立即调用 EEPROM_TypeB_eraseGroup。用户可以根据应用的要求选择合适的时间点进行擦除。

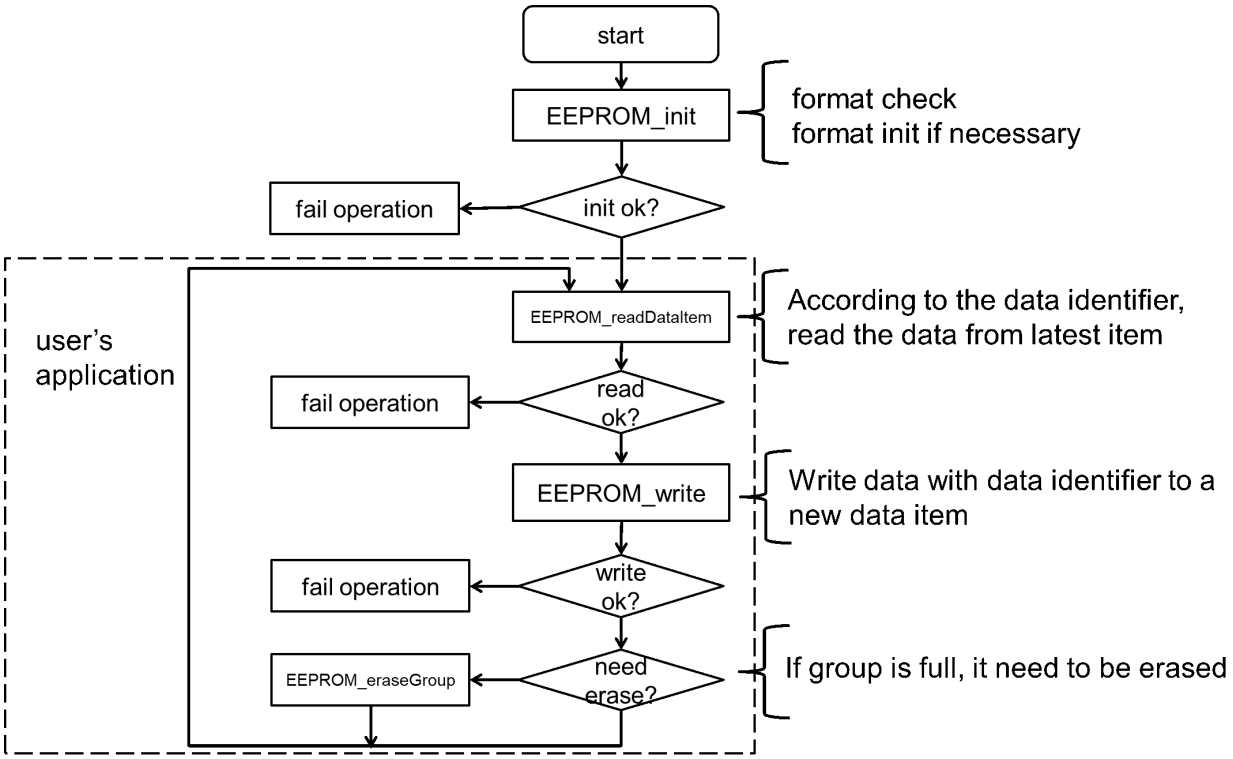

图 3-1. 概要软件流程

3.2 EEPROM 函数

实现此功能需要六个函数。四个全局变量用于记录 EEPROM 仿真的状态。

3.2.1 全局变量

有两个全局变量用于跟踪活动组。

- uint16_t gActiveDataItemNum;
- uint16_t gActiveGroupNum;

gActiveDataItemNum 用于记录数据项的数量。

gActiveGroupNum 用于记录活动组。

有两个全局变量用于表示标志。

- bool gEEPROMTypeBSearchFlag;
- bool gEEPROMTypeBEraseFlag;

当 EEPROM_TypeB_readDataItem 根据输入标识符找到数据项时，会设置 gEEPROMTypeBSearchFlag。

当组已满且需要擦除时，会设置 gEEPROMTypeBEraseFlag。

所有全局变量都在 eeprom_emulation_type_b.c 中定义。

3.2.2 EEPROM_TypeB_readDataItem

EEPROM_TypeB_readDataItem 用于读取与输入标识符匹配的数据项。图 3-2 中显示了软件流程。该函数会调用 EEPROM_TypeB_findDataItem 来查找数据项。

该函数的输入为标识符。该函数的输出为数据。此外，还会使用 gEEPROMTypeBSearchFlag 来显示是否找到数据项。

- 输入：uint16_t 数据标识符
- 输出：uint32_t 数据

图 3-3 显示了 EEPROM_TypeB_readDataItem 的不同情况。如果找到数据项，该函数将返回数据，并设置 gEEPROMTypeBSearchFlag。否则，该函数将返回 0，并清除 gEEPROMTypeBSearchFlag。通过检查该标志，用户可以判断是否找到数据并成功进行读取。

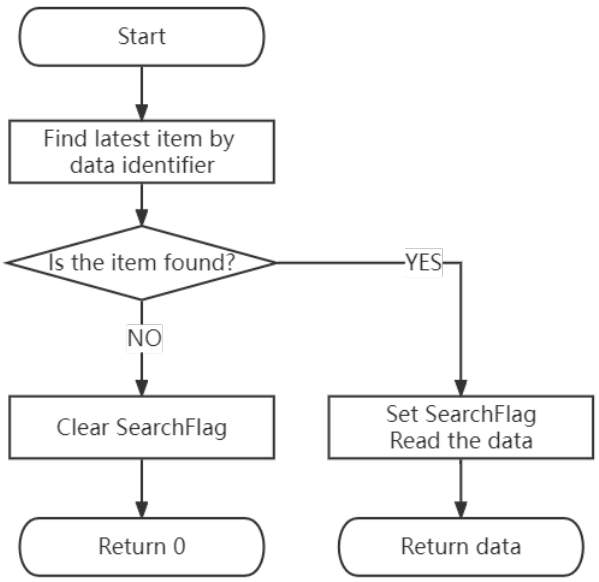

图 3-2. EEPROM_TypeB_readDataItem 的软件流程

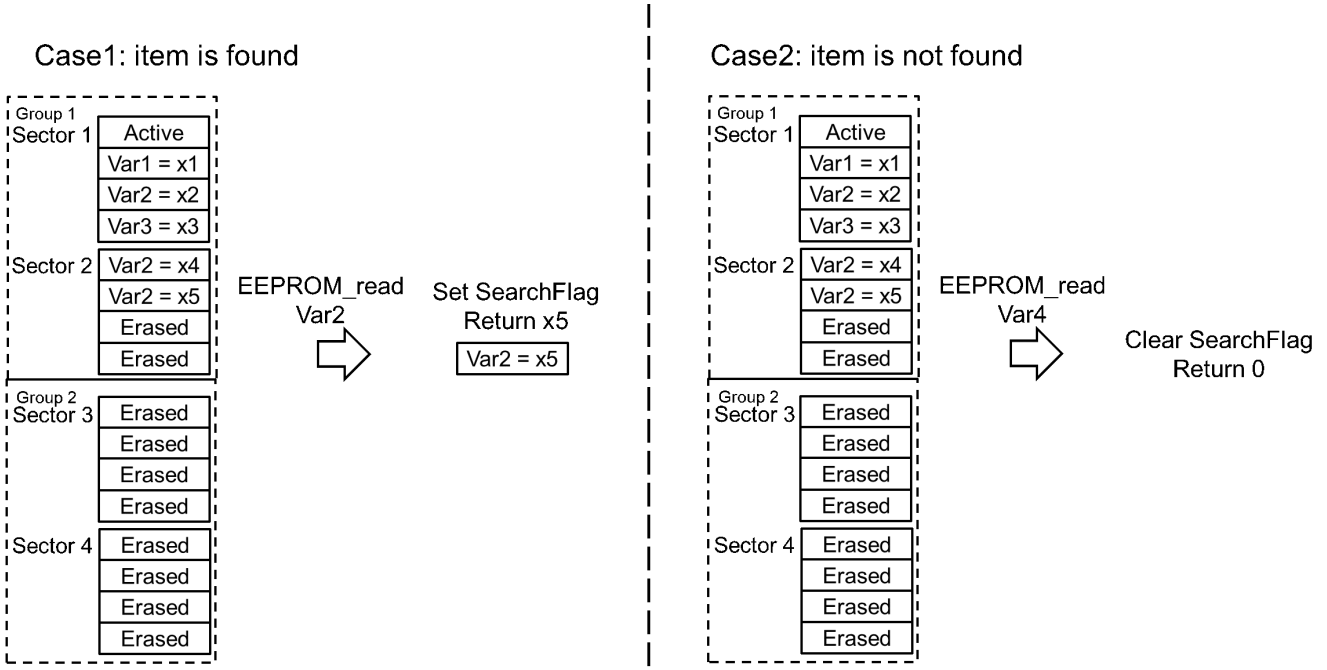

图 3-3. EEPROM_TypeB_readDataItem 的不同情况

3.2.3 EEPROM_TypeB_findDataItem

EEPROM_TypeB_findDataItem 用于搜索指定组内的数据项。图 3-4 中显示了软件流程。搜索过程从后往前遍历分组中的数据项，因此第一个找到的与标识符匹配的数据项一定是最新的数据项。

该函数的输入为标识符、GroupNum 和 DataItemNum。GroupNum 指定要搜索的组。DataItemNum 指定要搜索的最大数据项数。该函数的输出为地址。如果找到数据项，该函数将返回数据项的地址。否则，该函数返回 EEPROM_EMULATION_FINDITEM_NOT_FOUND（值为 0）。

- 输入：uint16_t 数据标识符

uint16_t GroupNum

uint16_t DataItemNum

- 输出：uint32_t 地址

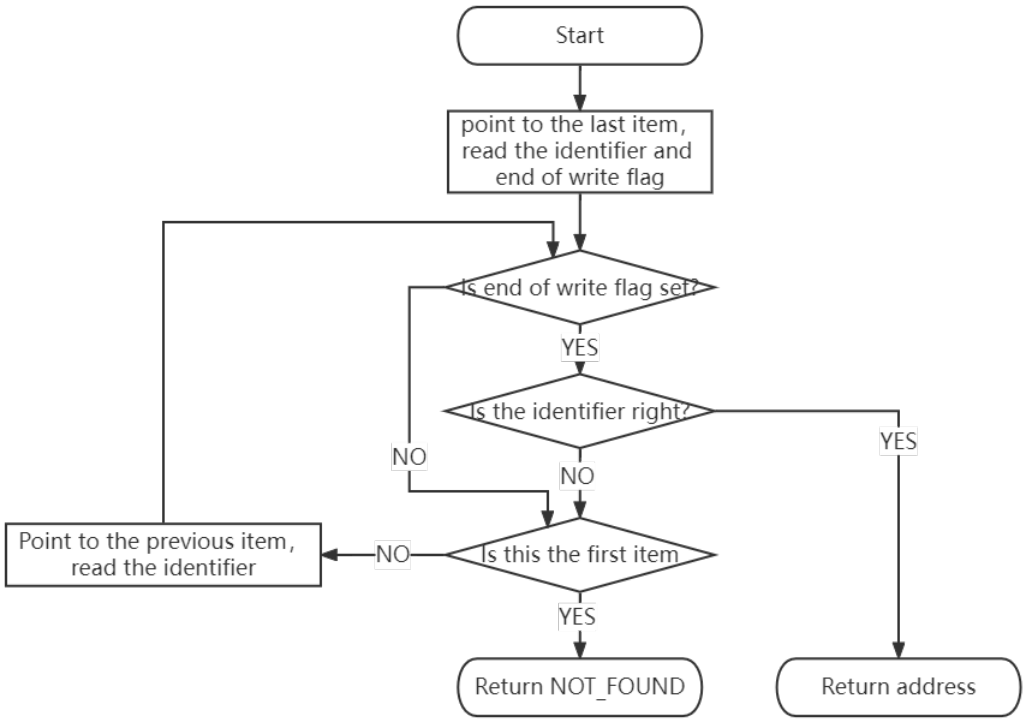

图 3-4. EEPROM_TypeB_findDataItem 的软件流程

3.2.4 EEPROM_TypeB_write

EEPROM_TypeB_write 用于将提供的数据和标识符写入闪存。通过该函数可以将新的数据项添加到闪存中。如果组已满，则会调用 EEPROM_TypeB_transferDataItem 并执行转移。图 3-5 中显示了软件流程。

首先，该函数检查下一个数据项是否被擦除。然后，它将数据和标识符汇集到一个新的数据项中，并设置写入结束标志以确保数据完整性。如果未设置写入结束标志，则数据项无效，所有函数都将跳过此项。

最后，该函数检查组是否已满。如果是，则执行转移。请参阅 EEPROM_TypeB_transferDataItem 以查看更多详细信息。

此方案允许用户选择要使用的标识符，但也要求用户注意使用的标识符数量。建议的标识符数量是最大数据项数的三分之一到一半。如果标识符数量接近数据项的最大数量，则会频繁进行转移和擦除，从而增加系统开销。如果标识符数量超过最大数据项数，则会导致错误。

该函数的输入为数据和数据标识符。该函数的输出为操作状态。此外，还会更新 gActiveGroupNum 和 gActiveDataItemNum 以跟踪活动组。

- 输入：uint32_t 数据

uint16_t 数据标识符

- 输出：uint32_t 操作状态

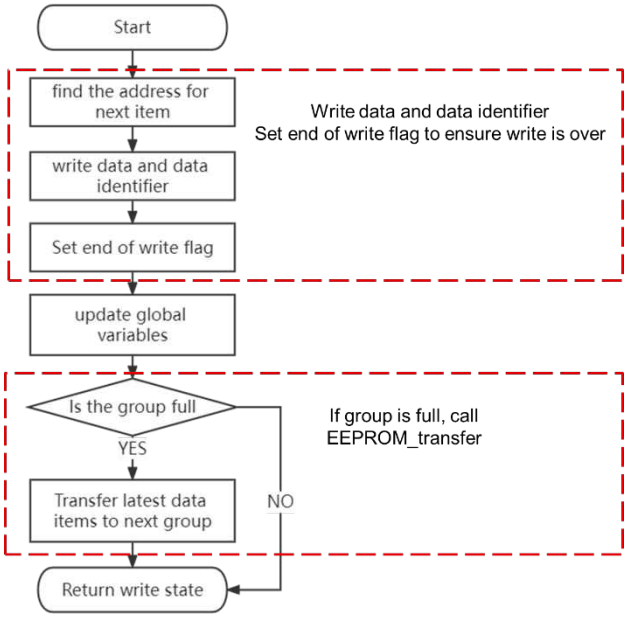

图 3-5. EEPROM_TypeB_write 的软件流程

3.2.5 EEPROM_TypeB_transferDataItem

EEPROM_TypeB_transferDataItem 用于将最新数据项从一个组转移到下一个组。并非所有项都将被转移。仅转移对应于每个标识符的最新数据项。

图 3-6 中显示了软件流程。首先，该函数会将下一个组更新为 *Receiving* 组。然后，它从后往前遍历当前组，检查数据项在 *Receiving* 组中是否已存在。如果不存在，则会转移该数据项。如果已存在，则跳过该数据项。转移后，最新数据项便已转移到 *Receiving* 组。最后，将 *Receiving* 组更新为 *Active* 组。转移出数据项的组会被标记为 *Erasing*，并会设置 gEEPROMTypeBEraseFlag。转移过程如图 2-2 所示。

通过检查 gEEPROMTypeBEraseFlag，用户可以调用 EEPROM_TypeB_eraseGroup 来擦除整个 *Erasing* 组。用户可以根据应用需求安排擦除时间点。

该函数的输入为 GroupNum，用于选择要转移的组。该函数的输出为操作状态。此外，还会更新 gEEPROMTypeBEraseFlag 来显示是否需要擦除。

- 输入：uint16_t GroupNum
- 输出：uint32_t 操作状态

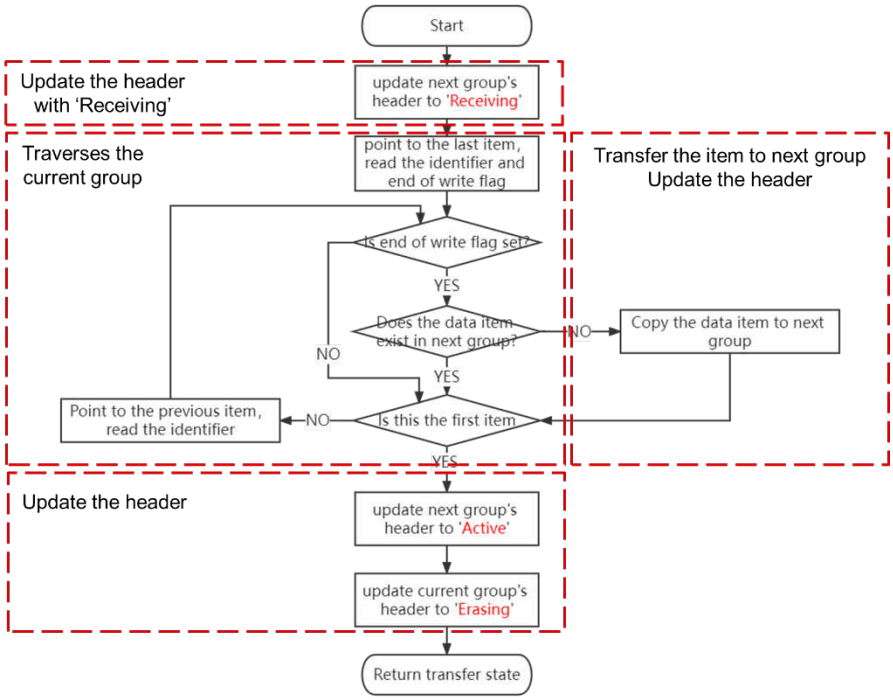

图 3-6. EEPROM_TypeB_transferDataItem 的软件流程

3.2.6 EEPROM_TypeB_eraseGroup

EEPROM_TypeB_eraseGroup 用于擦除 *Erasing* 组。调用 EEPROM_TypeB_transferDataItem 后将设置 gEEPROMTypeBEraseFlag。它可以提示用户存在 *Erasing* 组。建议在设置 gEEPROMTypeBEraseFlag 后立即调用 EEPROM_TypeB_eraseGroup，如图 3-1 所示。但是，用户可以通过修改概要软件流程来更改要擦除的时间点。

该函数的输出为操作状态。

- 输入：void
- 输出：uint32_t 操作状态

3.2.7 EEPROM_TypeB_init

该函数用于初始化 EEPROM 仿真。在使用 EEPROM 仿真之前，例如在器件上电后，应完成一次初始化。这样可以确保正确格式化相关的闪存区域并正确分配全局变量。

图 3-7 中显示了软件流程。首先，该函数会搜索活动组并通过遍历所有组标头来检查格式。如果存在活动组，它将擦除其他组并转移活动组。如果活动组不存在，则会擦除所有组。

图 3-8 和图 3-9 显示了 EEPROM_TypeB_init 的不同情况。A 是正常情况。B 是转移期间断电后的初始化。C 是未擦除 *Erasing* 组的情况。D 是所有组都为空的情况。E 是包含无效数据的情况。

该函数的输出为操作状态。此外，还会更新 gActiveGroupNum 和 gActiveDataItemNum 以跟踪活动组。

- 输入：void
- 输出：uint32_t 操作状态

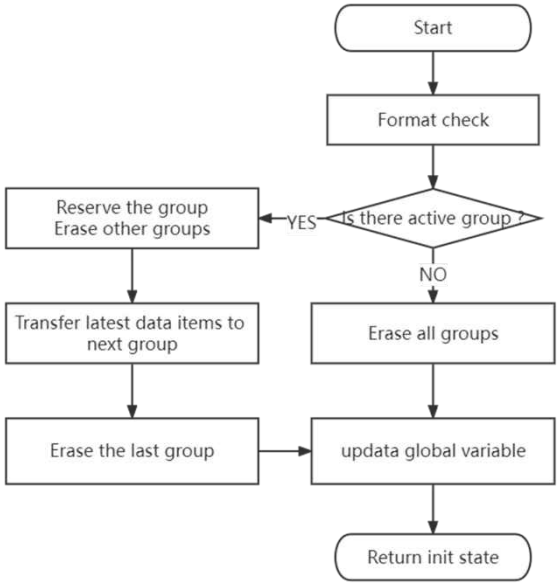

图 3-7. EEPROM_TypeB_init 的软件流程

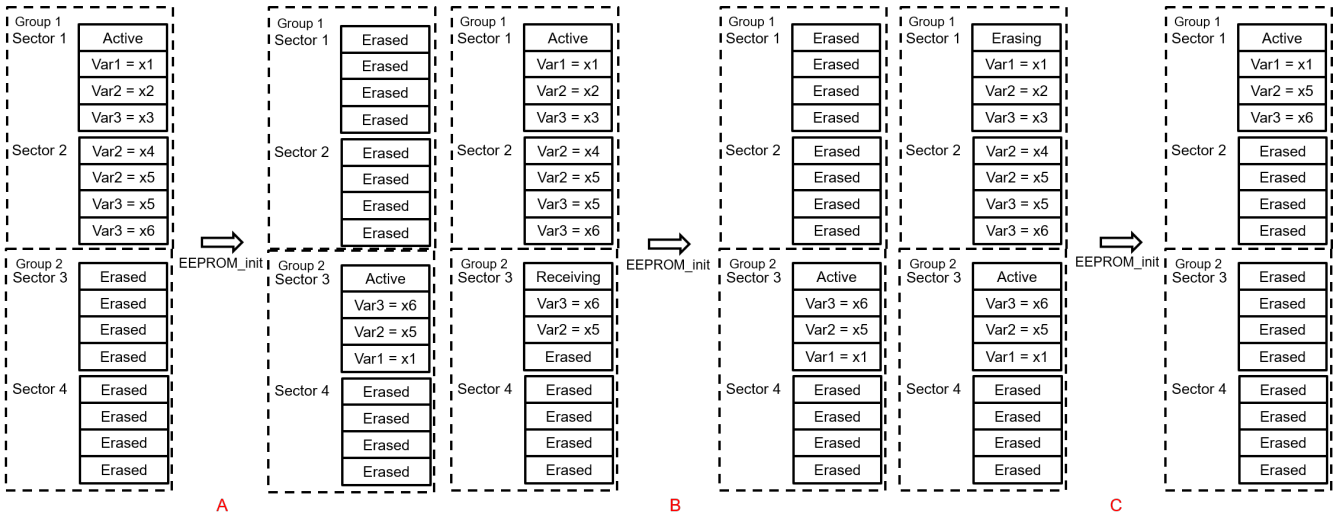

图 3-8. 存在活动组时，EEPROM_TypeB_init 的三种情况

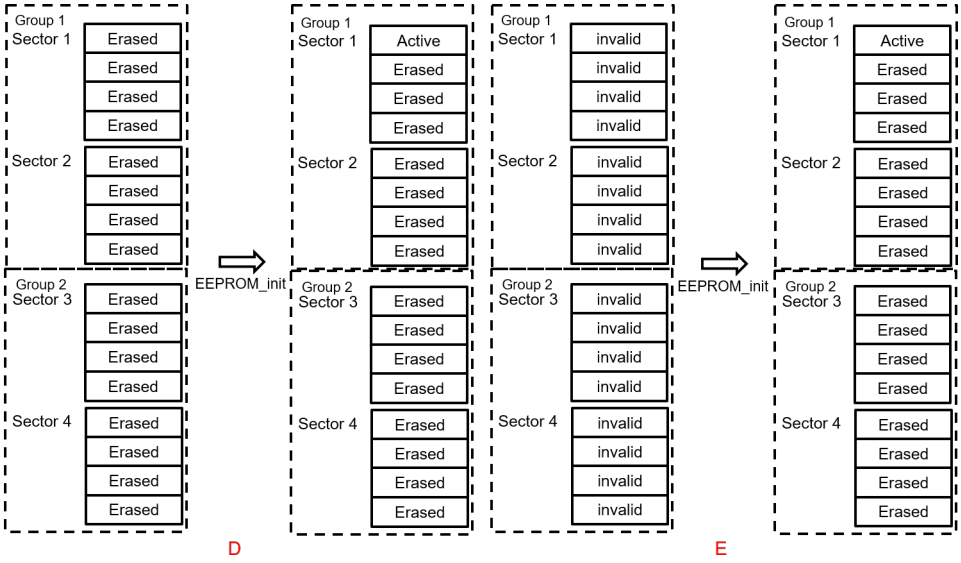

图 3-9. 不存在活动组时，EEPROM_TypeB_init 的两种情况

3.3 应用集成

需要此功能的应用程序必须包含为 MSPM0 MCU 提供的 eeprom_emulation_type_b.c 和 eeprom_emulation_type_b.h 文件。另外，还需包含面向特定器件的闪存 API。例如，对于 MSPM0G3507/MSPM0L1306，需要包含以下文件：

- eeprom_emulation_type_b.c
- eeprom_emulation_type_b.h
- ti_msp_dl_config.c
- ti_msp_dl_config.h

支持 MSPM0 产品的 SDK 中已包含 EEPROM 仿真库。

该 SDK 中还包含所有闪存 API 文件。

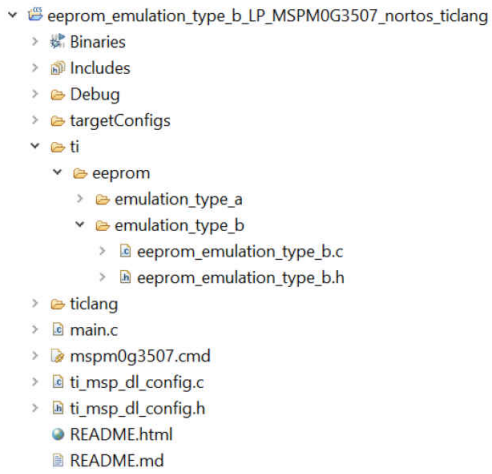

图 3-10. 软件所需的文件

3.4 EEPROM 仿真存储器占用空间

表 3-1 详细说明了 EEPROM 仿真驱动程序在闪存大小和 RAM 大小方面的占用空间。表 3-1 中的数据是使用 Code Composer Studio（版本：11.2.0.00007）且优化级别为 2 的条件下确定的。

表 3-1. EEPROM 仿真的结构

机制	所需的最小代码大小（字节）	
	闪存	SRAM
EEPROM 仿真 B 型	2816	6

3.5 EEPROM 仿真时序

本节介绍了与基于四个 1KB 闪存扇区的 EEPROM 仿真驱动程序相关的时序参数。执行所有时序测量的条件如下：

- MSPM0G3507
- 系统时钟为 32MHz
- 从闪存执行
- 室温

进行功能测试的参数如下：

- 组数：2
- 一个组中的扇区数：2
- 扇区地址：0x00001000

表 3-2. EEPROM 仿真操作时序

操作	最小值	典型值	最大值
EEPROM_TypeB_readDataItem	2.2us		89.4us
有活动组的 EEPROM_TypeB_init	4.65ms		61.38ms
无活动组的 EEPROM_TypeB_init		4.26ms	
无转移的 EEPROM_TypeB_write		109.5us	
有转移的 EEPROM_TypeB_write	630.8us		39.09ms
EEPROM_TypeB_eraseGroup		12.2ms	

4 应用方面

本节介绍了 EEPROM 仿真解决方案的应用级特性以及如何对其进行配置以满足应用需求。

4.1 可配置参数的选择

eeprom_emulation_type_b.h 中有三个用户可配置的参数。可根据应用程序的要求相应地配置这些参数。

- 组数：至少 2 个
- 一个组中的扇区数：至少 1 个
- 扇区地址

仅配置组数时，可以在图 4-1 中看到变化。当仅配置一个组中的扇区数时，可以在图 4-2 中看到变化。这两种配置方法都将增加所用的闪存区域，从而增加耐写次数。但配置组数时，最大数据项数不会改变。配置一个组中的扇区数时，最大数据项数将相应地改变。

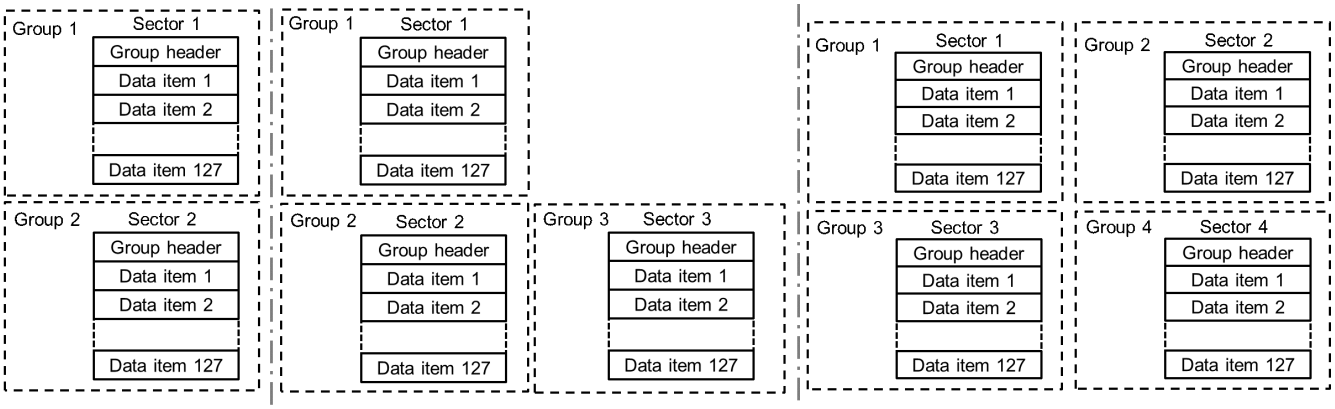

图 4-1. 仅配置组数时的结构变化

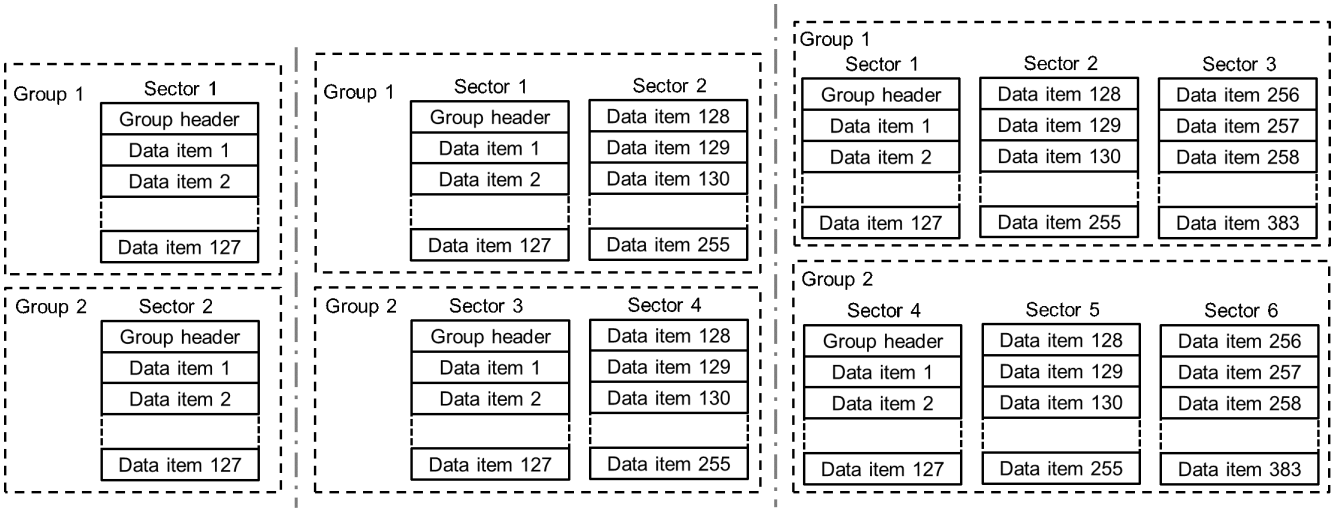

图 4-2. 仅配置一个组中的扇区数时的结构变化

4.1.1 数据项数

数据项的数量与组中的扇区数量直接相关。

$$\text{Number of data items} = \frac{\text{Sector size}}{\text{Data item size}} \times \text{Number of sectors in one group} - 1$$

选择正确数量的数据项至关重要。关键是根据应用需要存储的变量数量进行选择。如果变量的数量接近数据项的数量，则在更新这些变量的值时会频繁进行转移。如果变量的数量远远少于数据项的数量，则意味着组的大小相对较大，并会在转移、擦除和搜索等操作中花费额外的时间。

建议的标识符数量是最大数据项数的三分之一到一半。

4.1.2 耐写次数

闪存耐写次数通常为 10000 次，远低于实际 EEPROM。EEPROM 仿真的一个特性是其耐写次数高于闪存。闪存扇区将划分为多个数据项并逐一写入数据，不需要每次写入时都进行擦除，而是只在写满之后才需要擦除。此外，通过使用多个闪存扇区，等效的耐写次数将进一步提高。

一个组中的扇区数量以及组的数量都会影响等效的擦除/写入次数。由于一个组中的扇区数量已经在上面确定，因此现在可以通过调整组数来获得合理的等效擦除/写入次数。

$$\text{Effective endurance} = \frac{\text{Number of data items}}{\text{user's data items}} \times \text{Number of groups} \times \text{flash endurance}$$

建议用户在选择合理的组数之前计算应用程序所需的耐写次数。

对于三个可由用户配置的参数，建议的设计流程如下：

1. 计算应用程序所需的数据项数，并选择一个组中的合理扇区数。
2. 计算应用程序所需的耐写次数，并选择合理的组数。
3. 选择合适的闪存地址。

例如，有一个应用程序需要每 10 分钟更新一次 EEPROM 中的数据，共有 20 个变量，并需要保证 10 年不间断服务能力。首先，每个组只需要 1 个扇区（127 个数据项）。其次，应用程序所需的耐写次数为 525600 次（10 年 × 365 天 × 24 小时 × 6 次/小时）。需要九组，等效的耐写次数为 571500 次。

4.2 断电恢复

如果在 EEPROM_TypeB_write 或 EEPROM_TypeB_eraseGroup 期间断电，则可能会损坏数据。

为了检测损坏并从中恢复，实现了 EEPROM_TypeB_init。应在上电后立即调用该函数。EEPROM_TypeB_init 会检查所有组的标头，以确认 EEPROM 仿真的数据存储是否正确。

在 EEPROM 仿真的结构中，标头可显示相应组的状态。一共有四种状态。上一节详细介绍了这四种状态之间的变化。

5 参考文献

1. 德州仪器 (TI)，*使用低内存 MSP430™ FRAM MCU 的 EEPROM 仿真* 应用手册

重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265