

概述

EEPROM（电可擦除可编程只读存储器）通常用于工业应用中存储可更新的数据。EEPROM 是一种永久非易失性存储器系统，用于电子设备在电源故障时存储和保留少量数据。

为了低成本，外部 EEPROM 可以用片上 Flash 代替，并采用特定的软件算法。本文档介绍了使用 SPC1169/SPD1179 片上 Flash 存储器来模拟 EEPROM 的机制并替代独立 EEPROM 的软件解决方案。该方案采用 Flash 中的三个 Page（每个扇区为一个 Page）来实现。EEPROM 模拟代码以对用户透明的方式填充 Page 并在 Page 之间交换数据。

目录

1	外部 EEPROM 和模拟 EEPROM 之间的主要区别	7
1.1	写访问时间的不同.....	7
1.2	擦除时间的不同.....	7
1.3	写方式的相似之处.....	8
2	实现 EEPROM 模拟	9
2.1	原理	9
2.2	应用案例	10
2.3	模拟 EEPROM 固件说明	11
2.4	模拟 EEPROM 时序	12
3	嵌入式应用方面.....	14
3.1	数据粒度管理	14
3.2	磨损均衡算法	14
3.3	Page Header 恢复.....	14
3.4	实时性考虑	15

图片列表

图 2-1: SPC1169/SPD1179 用于模拟 EEPROM 的 Flash 扇区.....	9
图 2-2: Flash 页和模拟 EEPROM 变量格式.....	10
图 2-3: 数据更新过程	11

SPIN
TROL

表格列表

表 1-1: 外部 EEPROM 和模拟 EEPROM 之间的区别	7
表 2-1: 变量虚拟地址	10
表 2-2: 模拟 EEPROM 固件 API 说明	11
表 2-3: 模拟 EEPROM 固件 API 返回值说明	12
表 2-4: 模拟 EEPROM 时序 (SYSCLK = 100MHz)	12

SPIN TROL

版本历史

版本	日期	作者	状态	变更
A/0	2023 年 3 月 10 日	黄恒方	Released	首次发布。

SPIN
TROL

术语或缩写

术语或缩写	描述

SPIN TROL

1 外部 EEPROM 和模拟 EEPROM 之间的主要区别

EEPROM 是许多嵌入式应用（需要进行非易失性数据存储，且运行期间以 Byte 或 Word 的颗粒度进行更新）的关键元件。

用于这些系统的微控制器通常基于嵌入式 Flash 存储器。为了减少所用元件，节省 PCB 空间和降低系统成本，SPC1169/SPD1179 的 Flash 存储器可以用来替代外部 EEPROM，进行代码和数据存储。

然而与 Flash 存储器不同的是，在数据被重新写入前，外部 EEPROM 不需要擦除操作来释放空间。需要专门的软件管理来将数据存入嵌入式 Flash。

模拟软件的机制取决于多种因素，包括 EEPROM 可靠性、Flash 存储器的结构以及产品需求。嵌入式 Flash 和外部串行 EEPROM 之间的主要区别对于任何使用同样 Flash 技术的微控制器（并非特指 SPC1169/SPD1179 产品）都是相同的。

表 1-1: 外部 EEPROM 和模拟 EEPROM 之间的区别

特性	外部 EEPROM（例如，AT24C02： I2C 串行访问 EEPROM）	利用片上 Flash 模拟 EEPROM
写操作时间	<ul style="list-style-type: none"> - 写一个 Byte: 5ms 以内。因此，写一个 Word = 20ms - 页（8Bytes）写: 5ms 以内。因此，写一个 Word = 2.5ms 	写一个 Word（32-bit）：从 44.3us 到 15.03ms
擦除时间	N/A	擦除 3 个 Page: 4275.16us
写方式	启动后，不依赖 CPU 只需要供电	一旦启动，则依赖 CPU
读访问时间	读一个 Word（32-bit）：66us	读一个 Word（32-bit）：2.18us （SYSCLK = 100MHz）
写/擦除周期	100 万次	每个 Flash 页 10 万次 使用多个片上闪存内存页相当于增加写入周期数。

1.1 写访问时间的不同

由于 Flash 的写访问时间更短，重要参数可以更快地存入模拟 EEPROM（比外部串行 EEPROM 更快），因此能够提高数据存储能力。

1.2 擦除时间的不同

擦除时间的不同是独立 EEPROM 和使用嵌入式 Flash 模拟的 EEPROM 之间的另一个重要区别。与 Flash 不同，在向其写入数据前，EEPROM 不需要擦除操作来释放空间。这意味着需要某种形式的软件管理来将数据存入 Flash。此外，由于 Flash 中的块擦除过程不需要花费很多时间，因此设计 Flash 管理软件时，应当考虑断电和其他可能中断擦除过程的事件（例如复位）。为了设计出稳健的 Flash 管理软件，有必要彻底了解 Flash 擦除过程。

注意：软件复位过程中，SPC1169/SPD1179 Flash 上正在进行的扇区擦除或批量擦除操作会被中断。

1.3 写方式的相似之处

外部 EEPROM 和采用 SPC1169/SPD1179 嵌入式 Flash 模拟的 EEPROM 之间的相同点之一是其写方式：

- 独立外部 EEPROM：一旦被 CPU 启动，写操作不会被软件复位中断。只有电源故障会中断写过程，因此，正确确定去耦电容的大小可以保护独立 EEPROM 内的完整写入过程。
- 利用嵌入式 Flash 模拟的 EEPROM：一旦被 CPU 启动，写过程可被电源故障中断。软件复位会中断 SPC1169/SPD1179 嵌入式 Flash 上正在进行的写操作。

2 实现 EEPROM 模拟

2.1 原理

考虑到 Flash 存储器本身的特性和产品需求，存在多种方式实现模拟 EEPROM。下面所描述的方式需要至少两个相同大小的 Flash 页（分配给非易失性数据）。

第一个 Flash 页最初被擦除并用于存储新数据，Flash 编程操作按照 Flash 地址的递增顺序依次完成。一旦第一个 Flash 页充满了数据，需要将第一个 Flash 页中的有效数据搬移到第二个 Flash 页。

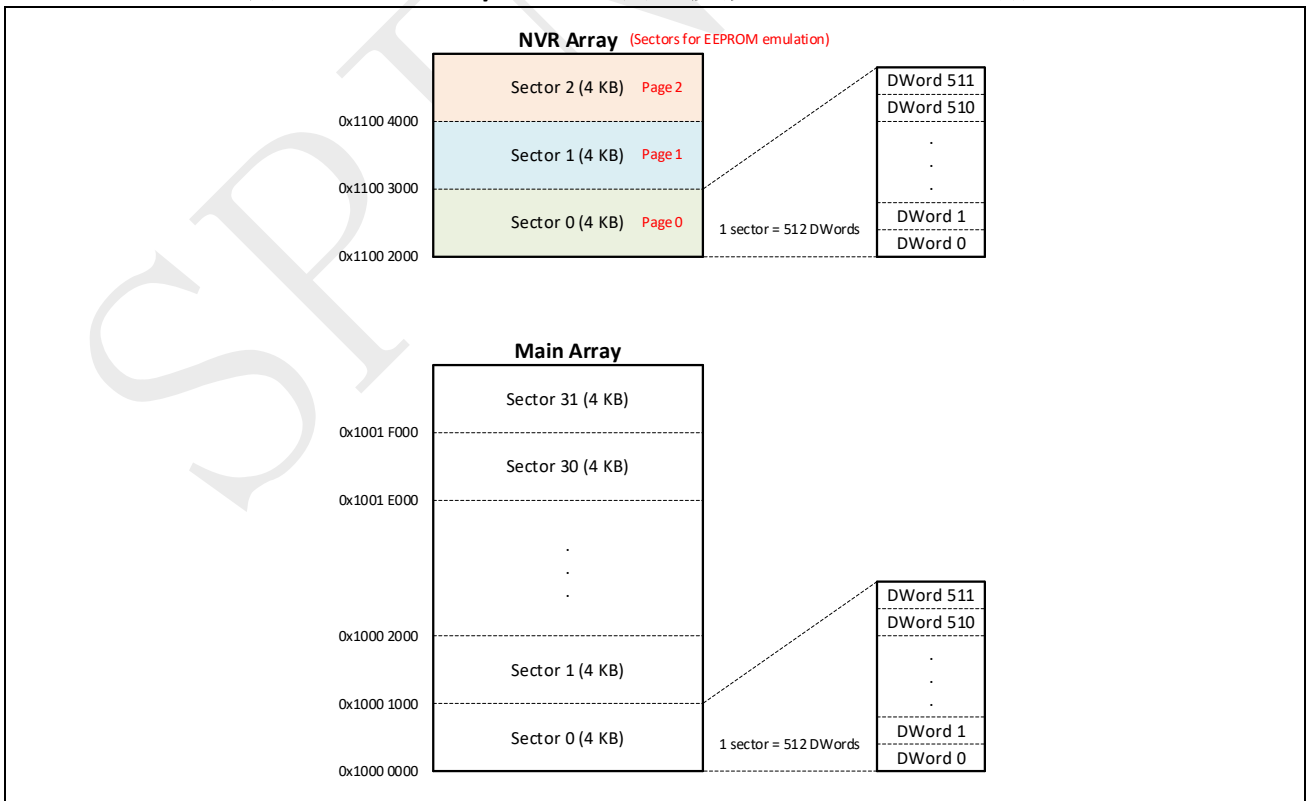
第二个 Flash 页仅收集第一个 Flash 页的有效数据，其余的区域可用于存储新数据。当将有效数据搬移到第二个 Flash 页之后，可以擦除第一个 Flash 页。

每个 Flash 页可以由 1 个或者多个 Flash 扇区组成。对于 SPC1169/SPD1179 而言，使用 3 个 Flash 页来实现 EEPROM 模拟，每个 Flash 页由 1 个 Flash 扇区组成，如所图 2-1 示。这 3 个 Flash 页分贝叫做 Page 0、Page 1 和 Page 2。

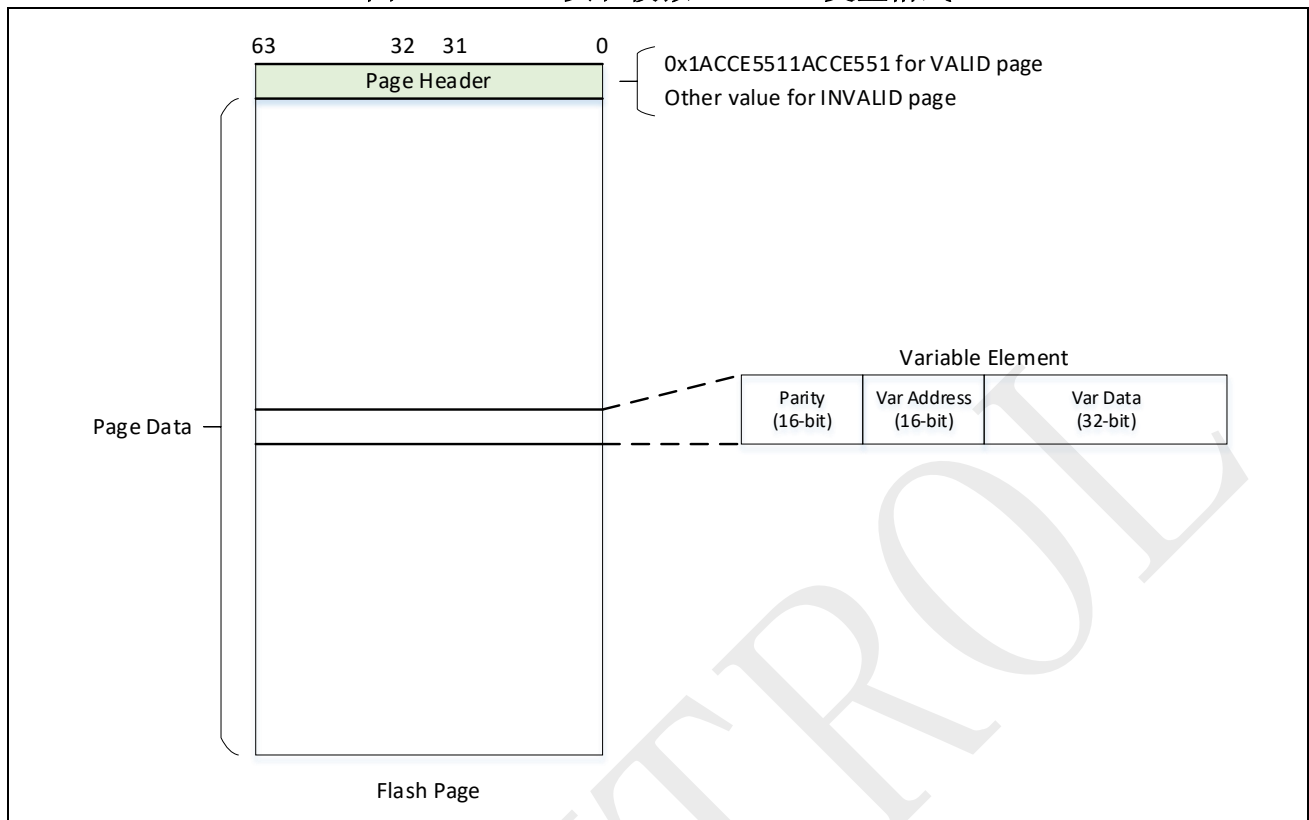
每个 Flash 页的第一个 64 比特的双字（Double-Word）作为 Page Header，标识当前 Flash 页的状态。每个 Flash 页有下面两种可能的状态：

- INVALID: 该页为空（初始状态，页擦除后即为空），或者正在接收其他页搬移来的数据；
- VALID: 该页包含有效数据，用于存储新数据；直到所有有效数据被搬移到另一个 Flash 页，改页状态才会改变。

图 2-1: SPC1169/SPD1179 用于模拟 EEPROM 的 Flash 扇区



[1] DWord = Double Word, 1 DWord = 64-bit.

图 2-2: Flash 页和模拟 EEPROM 变量格式


每个变量元素（Variable Element）都由一个虚拟地址（Var Address）和一个要存储在 Flash 中的数据值（Var Data）来定义，该数据值用于后续的检索或者更新。虚拟地址为 16 位宽，数据值为 32 位宽。SPC1169/SPD1179 模拟 EEPROM 固件要求的虚拟地址范围为 0 ~ 255。

每个变量元素还包含一个 16 位 Parity，用于检查变量元素的完整性。修改数据时，与同一虚拟地址关联的修改后数据存储在新的 Flash 位置。数据检索会返回最新的数据值。

2.2 应用案例

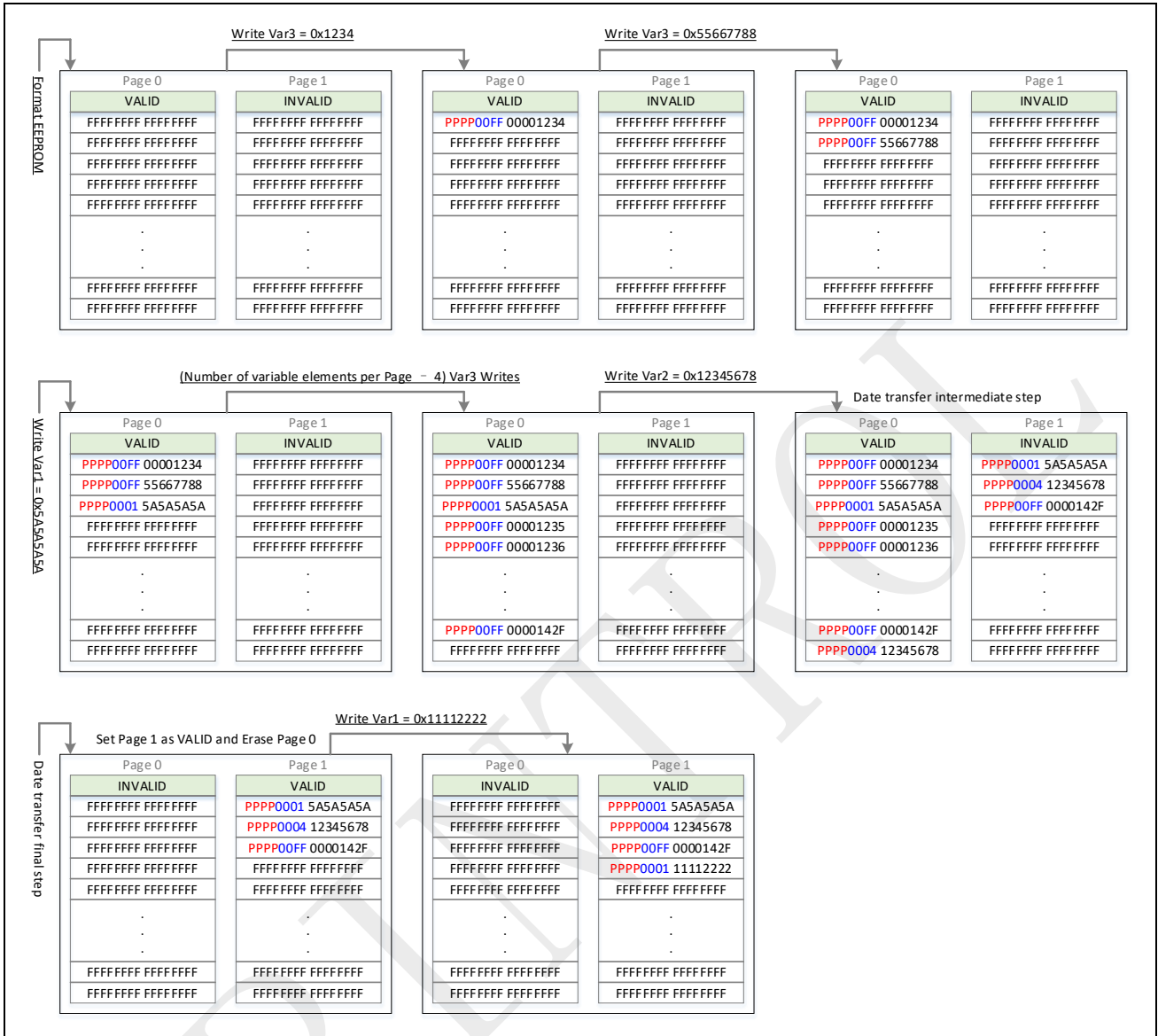
下面的示例展示了 3 个 EEPROM 变量的软件管理。变量的虚拟地址如表 2-1 所示。

表 2-1: 变量虚拟地址

变量	虚拟地址	数据位宽
Var1	0x01	32 位
Var2	0x04	32 位
Var3	0xFF	32 位

在本例中，只用到了两个 Flash 页：Page 0 和 Page 1。图 2-3 中，只展示了按 Flash 地址递增顺序执行的写命令，其中，PPPP 代表变量元素的 16 位 Parity。

图 2-3: 数据更新过程



2.3 模拟 EEPROM 固件说明

应用程序可以调用的模拟 EEPROM 固件函数如表 2-2 所示。

表 2-2: 模拟 EEPROM 固件 API 说明

函数名称	说明
EEPROM_Init	将模拟 EEPROM 变量元素配置为其初始状态，并将 Flash 页恢复到已知的良好状态。 芯片复位后，在访问模拟 EEPROM 之前，应该首先调用该函数。
EEPROM_Format	擦除所有用于模拟 EEPROM 的 Flash 页，并将 Page 0 设置为 VALID
EEPROM_ReadWord	该函数用于获取虚拟地址对应的变量元素(最近一次存储)的数据值。 若找不到虚拟地址对应的变量元素，该函数返回 EEPROM_STATUS_NO_DATA。
EEPROM_WriteWord	该函数用于更新虚拟地址对应的变量元素的数据值。

表 2-3: 模拟 EEPROM 固件 API 返回值说明

返回值符号	返回值	描述
EEPROM_STATUS_OK	0x00000000U	操作成功
EEPROM_STATUS_WRITE_ERROR	0x00000001U	Flash 编程操作错误
EEPROM_STATUS_ERASE_ERROR	0x00000002U	Flash 擦除操作错误
EEPROM_STATUS_INVALID_ADDR	0x00000004U	虚拟地址参数值不在有效范围(0~255)内
EEPROM_STATUS_WRITE_CHECK_FAIL	0x00000008U	写操作时, 写入的变量元素回读校验错误
EEPROM_STATUS_NO_DATA	0x00000010U	读操作时, 模拟 EEPROM 中找不到虚拟地址对应的变量元素
EEPROM_STATUS_INVALID_ENTRY	0x00000020U	写操作时, 要写入变量元素的 Flash 地址, 不在当前有效 Flash 页的地址范围内
EEPROM_STATUS_ELEMENT_NOT_EMPTY	0x00000040U	写操作时, 要写入变量元素的 Flash 地址内容不为空
EEPROM_STATUS_ADDR_MISMATCH	0x00000080U	读操作时, 虚拟地址参数值与读取的模拟 EEPROM 变量元素中的虚拟地址不一致
EEPROM_STATUS_PARITY_ERROR	0x00000100U	读操作时, 读取的模拟 EEPROM 变量元素 Parity 检验错误
EEPROM_STATUS_NO_PAGE_FOUND	0x00000400U	读/写操作时, 未找到 VALID 的 Flash 页
EEPROM_STATUS_PAGE_HEADER_ERROR	0x00000800U	设置 Flash 页 Page Header 时发生错误
EEPROM_STATUS_TRANSFER_ERROR	0x40000000U	Flash 页之间搬移数据错误, 与其他错误返回值逻辑或
EEPROM_STATUS_INVALID_HEADER	0x80000000U	无效的 Flash 页 Page Header 状态组合 此时, 需要调用 EEPROM_Format 函数格式化模拟 EEPROM

2.4 模拟 EEPROM 时序

本章节介绍 SPC1169/SPD1179 模拟 EEPROM 固件相关的时序参数, 如表 2-4 所示。

表 2-4: 模拟 EEPROM 时序 (SYSCLK = 100MHz)

操作	模拟 EEPROM 时序		
	最小	典型	最大
EEPROM_WriteWord (典型情况)	-	44.3 us	-
EEPROM_WriteWord (伴有 Flash 页数据搬移情况)	-	15.03 ms	-
EEPROM_Read	-	2.18 us	-
EEPROM_Format (三个 Flash 页均为 INVALID 且非空)	-	11.68 ms	12.63 ms

操作	模拟 EEPROM 时序		
	最小	典型	最大
EEPROM_Init (典型情况, 一个 Flash 页 VALID, 另外两个 Flash 页为空)	-	626 us	-
EEPROM_Init (典型情况, 一个 Flash 页 VALID, 另外两个 Flash 页 INVALID 且非空)	-	7.99 ms	8.98 ms

SPIN TROL

3 嵌入式应用方面

本章节提供有关如何克服嵌入式应用程序中的软件限制以及如何满足不同应用程序需求的建议。

3.1 数据粒度管理

模拟 EEPROM 可用于需要以字节、半字（Half-Word）或字（Word）粒度更新数据的非易失性存储的嵌入式应用。数据大小通常取决于应用要求，例如传感器或通信接口数据大小。

模拟 EEPROM 固件设计是用于支持 32 位字粒度。对于字节、16 位半字的非易失性存储需求，应用程序可以先将其转换成 32 位字类型，然后再通过模拟 EEPROM 固件存储到模拟 EEPROM 中。但是，这种方式对 Flash 的存储空间利用率较低。为了优化 Flash 的使用，用户应用程序可以将所有较小尺寸的数据元素收集到 32 位数据元素中，然后再将内容存储在模拟 EEPROM 中。这将通过同时写入 16 位虚拟地址、16 位 Parity 和 32 位数据值来确保 64 位 Flash 的最佳使用。

3.2 磨损均衡算法

磨损均衡算法允许监控和均匀分布不同 Flash 页之间的写和擦除操作。当不使用磨损均衡算法时，Flash 页不会以相同的频率使用。例如，具有长期数据的 Flash 页不会像包含频繁更新数据的 Flash 页那样经历那么多的写和擦除周期。磨损均衡算法可确保平等地使用每个 Flash 页的所有可用的写周期。

根据设计，EEPROM 模拟算法在 Flash 页之间均匀分配 Flash 写和擦除操作。无论什么用户变量被写入，Flash 写操作都按地址递增的顺序执行。当一个 Flash 页已满时，有效变量元素将被复制到另一个 Flash 页，并且第一个 Flash 页将被完全擦除。磨损均衡算法具体实现如下：

- 当 Page 0 已满时，将 Page 0 中的有效变量元素复制到 Page 1 中并将 Page 1 设置为 VALID，然后擦除 Page 0；
- 当 Page 1 已满时，将 Page 1 中的有效变量元素复制到 Page 2 中并将 Page 2 设置为 VALID，然后擦除 Page 1；
- 当 Page 2 已满时，将 Page 2 中的有效变量元素复制到 Page 0 中并将 Page 0 设置为 VALID，然后擦除 Page 2；
- 依此类推。

3.3 Page Header 恢复

在 Page Header 更新、Flash 页擦除或者数据搬移期间断电或复位的情况下，Page Header 可能会损坏。为了检测该损坏并从中恢复，模拟 EEPROM 固件提供了 EEPROM_Init 函数。上电后，应当立即调用该函数，检查 Flash 页的状态和完整性，并在必要时进行修复。这可确保不会丢失任何数据。

为了避免可能的故障情况，在复位后，EEPROM_Init 函数会检查已经擦除的 Flash 页是否为空，如果不为空，会再次擦除该 Flash 页。

3.4 实时性考虑

SPC1169/SPD1179 的模拟 EEPROM 固件运行在内部 Boot ROM 中。在调用模拟 EEPROM 固件函数进行 EEPROM 初始化、变量元素更新或者 Flash 页擦除期间，CPU 对 Flash 的访问会停止，无法执行 Flash 中的应用程序。对于许多应用程序，该行为可能是可以接受的，但是对于具有实时性约束的应用程序，用户需要采取改进措施，如将有实时性要求的相关代码放在 RAM 中运行。

SPIN
TROL