文章编号:1001 - 9081(2006)05 - 1205 - 03

一种 Flash 存储器静态负载平衡策略

张 骏[1],樊晓桠[1],刘松鹤[2]
(1. 中国航空工业总公司 航空微电子技术研究与培训中心,陕西 西安 710072;
2. 长安大学 信息工程学院,陕西 西安 710064)
(freakfriend@163.com)

摘 要:Flash 存储器是常用的存储器件,它的不挥发特性、灵活轻便、低功耗特性等特点为嵌入式系统、移动计算机提供了良好的条件。然而,Flash 存储器必须先擦除后写入,这对存储系统的设计提出了挑战。更重要的是,它的擦除次数是有限制的。分析了 Flash 存储器数据存储结构和物理特性,提出了一种触发式数据分类静态负载平衡策略,并在此基础上作了基于 VHDL 语言的仿真,收到了较好的效果。

关键词:Flash;静态负载平衡;仿真
中图分类号: TP303 **文献标识码**:A

Static wear-leveling strategy for flash memory

ZHANG Jun[1], FAN Xiao-ya[1], LIU Song-he[2]
(1. *Micro-electronics Research and Training Center, China Aviation Industry Corporation, Xi'an Shaanxi* 710072, *China;*
2. *College of Information Engineering, Changan University, Xi'an Shaanxi* 710064, *China*)

Abstract: Flash Memory is a kind of common storage device. Its characteristics of flexibility, low power, and so on offer excellent qualifications for embedded system and mobile system. But Flash Memory must be wrote after erasure operation, which brings forward challenge for storage system designer. The most important thing is that the erasure operation times are very limitable. The data structure and physical characteristics were analysed. And a static wear-leveling strategy based on classifying data with trigger condition was brought forward. An experiment was carried out to simulate this strategy using VHDL. As a result, this strategy improves the wear-leveling rate.

Key words: Flash; static wear-leveling; simulation

Flash 存储器有高速的数据存取速度、体积小、功耗低、轻便、抗震,更重要的是即使系统掉电它的数据也不会丢失。但是 Flash 存储器有独特的硬件特性。要改写块内的内容,即使是一个扇区,也要先进行整个块的擦除操作,然后再进行新数据的写入。这是很耗时、耗能的,并且会使系统的性能下降。另外,写操作又比读操作要慢,而擦除操作比写操作还要慢很多,更糟的是对同一个数据块的擦除次数是有限制的,超出规定的擦除次数某些块就会提前损坏。所以,在实际应用中 Flash 存储器的驱动程序中都有负载平衡策略。但是这些策略绝大多数都是动态负载平衡策略,重点放在写入数据时对目标扇区的选择上,而关于静态负载平衡策略的研究却很少。如果 Flash 存储介质中所含有的静态数据不断增加,那么动态负载平衡策略的应用范围就会越来越小,最终会导致某些数据块提前损坏。静态负载平衡的研究重点就是如何使静态数据也能在整个存储介质上"运动"起来,从而使动态负载平衡策略在介质内含有大量静态数据的情况下也能应用于整个存储介质,进而达到延长存储器寿命的目的。

1 相关的工作

对 Flash 存储器管理策略的研究大都集中在以下几个方面:采用什么样的动态写策略更有效;如何以更小的功耗和更少的操作为代价来回收更多的空间;如何根据数据状态信息来形成更合理的 Flash 存储器数据存放结构,从而获得更高的性能;但对于静态负载平衡方面的研究却非常的缺少。

文献[1]建立了一个 Flash 存储器服务器,并通过在内存中建立 Flash 数据单元的详细信息表格,从而在写操作时能更加准确的找到合适的空闲目标扇区,从而提高了动态负载平衡的性能。同时还提出了一种新的基于清理代价的数据清理策略。文献[4]在较低的数据清理代价和较好的负载平衡性能之间作了一个平衡,通过分离不同属性数据块的方法来提高动态负载平衡和数据清理的性能。文献[2,3]中还对不同的数据清理策略做了研究,比较了 GREEDY,CBP,CAT 等几种数据清理算法的优点和不足。

归根结底,这些策略的使用都是为了减少对 Flash 存储介质的擦除操作,从而延长 Flash 存储器的寿命。更进一步的说,是为了使所有可擦除数据块上的擦除次数尽可能的平衡,从而避免了某一个数据块提前损坏。好的静态负载平衡策略能扩大动态负载平衡策略应用的范围,防止由于大量静态数据的存在而使动态负载平衡受到局限。静态负载平衡操作过后往往留下大量的无效数据,所以还需要配合较好的数据清理策略,这样才能收到较好的效果。

2 Flash 数据结构及其相关信息

图 1 是 Flash 存储器的数据结构[2,3]。Flash 存储器介质被分成许多物理上可擦除的块,每个数据块都包含一个数据块头,其中包含着本数据块的必要信息。例如本数据块已经

收稿日期:2005 - 11 - 14;**修订日期**:2006 - 02 - 17 **基金项目**:国家"十五"预研基金资助项目(41308010307)

作者简介:张骏(1978 -),男,陕西西安人,博士研究生,主要研究方向:先进微处理器体系结构; 樊晓桠(1962 -),男,湖南人,教授,博士生导师,主要研究方向:专用集成电路、计算机系统结构; 刘松鹤(1979 -),女(蒙古族),吉林松原人,硕士研究生,主要研究方向:计算机软件、计算机网络.

被擦除的次数、时间标记、数据有效标记、清理标记和本数据块中每个扇区的信息。每个扇区的信息又包括逻辑扇区号、时间标记、已经被更新的次数、数据有效和数据无效标志。这些信息是进行静态负载平衡的必要条件。另外,还有一块空间用来存储全局信息,包括 Flash 存储器所包含的数据块总数和扇区总数。

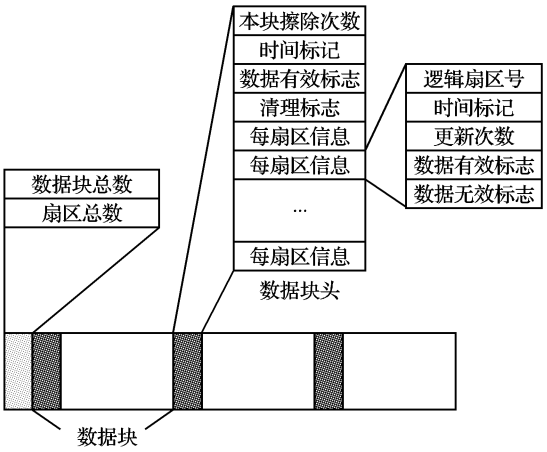
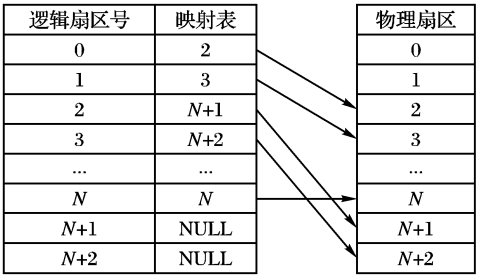


图 1 Flash 存储器的数据结构

由于在负载平衡的过程中逻辑扇区和物理扇区的一一对应关系被打乱,所以要正确存取数据就必须在内存中建立一个物理扇区号和逻辑扇区号的空间映射表,使逻辑扇区与物理扇区号进行动态的匹配。这样,在修改某个扇区的内容时只要空间映射表的内容将数据写入一个新的扇区,然后将原扇区标记为无效就可以了。在启动或者使用 Flash 存储器的时候,系统会加载 Flash 存储器驱动程序,由驱动程序收集每个数据块头信息中的逻辑扇区号,然后在内存中建立空间映射表。如图 2 所示:



逻辑扇区号	映射表
0	2
1	3
2	*N*+1
3	*N*+2
...	...
N	*N*
N+1	NULL
N+2	NULL

物理扇区
0
1
2
3
...
N
N+1
N+2

图 2 Flash 存储器空间映射表

扇区信息查找表[2,3]的创建过程与空间映射表相同,它用来记录每个扇区的相关信息(例如每个扇区的擦除次数、时间标记和一些控制标志)。由于是建立在内存中,在应用静态负载平衡策略的时候,根据扇区信息查找表中的内容就能快速的定位最佳的目标扇区。如图 3 所示。

逻辑扇区号	擦除次数	数据有效标志	数据无效标志	时间标记	坏扇区标记
1					
2					
...	...	...	...	...	...
i -1					
i					

图 3 扇区信息查找表

3 Flash 存储器的静态负载平衡

所谓负载平衡就是使用 Flash 存储器作为存储系统时要尽量避免对同一个数据块进行多次擦除操作,要使所有数据块的擦除次数尽可能的平衡。这样才能延长 Flash 存储器的寿命,同时也会提高系统性能。

Flash 存储介质上可能包括静态文件,它的特点是有成块的数据在很长的一段时间内是不变化的,有的甚至和闪存存储器的寿命一样长。如果负载平衡只应用于那些新写入的扇区,那么静态区域将永远也得不到写周期。这样一来,如果闪存中含有大量的静态数据,Flash 存储器的寿命就会大大降低。要解决这个问题,就要有相应的静态负载平衡策略强迫数据像在动态区域中一样在静态区域上传输,这样就能在整个存储介质上应用负载平衡了。

有两个关键的因素决定着静态负载平衡策略的优劣:一个是什么时候进行静态负载平衡操作,也就是静态负载平衡的条件;另一个是对静态数据所移动到的目的扇区的选择。根据 Flash 存储器的空间映射表和扇区信息查找表可以更精确的确定这两个条件。

3.1 静态负载平衡的触发条件

根据以上所述,作者提出了一种触发式数据分类静态负载平衡策略。利用在内存中建立好的扇区信息查找表对各个数据有效的扇区进行监控,并动态地确定一个以各扇区的擦除次数和时间标记为基础的触发条件。假设触发条件为 K,n 为总的扇区数目,M 为每个扇区的时间标记,T 为每个扇区的擦除次数,比例系数为 β,则这个触发条件可以用式(1)表示:

$$K = \beta \times \frac{\sum_{0}^{n-1} T_i}{n \times M} \tag{1}$$

驱动程序设计者可以根据系统的需要通过对比例系数 β 进行修改来改变触发阈值的高低。每次写扇区操作到来的时候用各扇区的擦除次数与时间标记之比和这个触发条件作比较。如果某一个扇区的擦除次数与时间标记之比低于 K 就把这个扇区中的内容搬移到当前擦除次数最高的可用空扇区中去,并将原扇区标记为数据无效,等待可用空扇区不够时对这些数据无效扇区进行空间回收;如果所有扇区的擦除次数与时间标记之比都比 K 高,那么就不进行搬移,这样静态数据就会进入到擦除次数已经很高的扇区上去,并将擦除次数较少的扇区空出来,从而实现静态负载平衡。这种策略是建立在假定静态数据在未来的一段时间内仍旧是静态的基础上的。

3.2 静态数据传输到目的扇区的选择

假设静态负载平衡操作已经被触发,下一步就是对静态数据所移动到的目的扇区进行选择。最简单的方法是随机选择一组擦除次数已经很高的空扇区来进行静态数据存储,但这对空间回收时的擦除操作没有任何减少。如果能将更新频率非常高的动态数据和静态数据进行分类,那么经常被更新的动态数据和不经常被更新的静态数据就会被分离而处在不同的数据区域中。这种数据区域是在逻辑上进行划分的,每个区域包含一个或者多个数据块。也就是说一个数据块中要么几乎都是动态数据,要么几乎都是静态数据。含有更新频率非常高的动态数据的数据块最有可能被更新,根据负载平衡策略,将有大量的无效数据被留在这个数据区域中。而擦除操作是针对整个数据块的,很显然,清理一次含有经常更新数据的数据块将得到更多的空扇区,也就是更高的回收率。所以需要进行清理的次数就会变少,清理过程中需要搬移的有效数据也会变少,这不但减少了擦除次数,并且降低了功耗。

以数据块的擦除次数为基础,作者提出了一种数据区域划分算法。过程如下:假定 T_{max} 和 T_{min} 分别为数据块更新次数的最大值和最小值。则在一个确定的时刻,T_{max} 和 T_{min} 所对应的数据块就是更新频率最大和最小的数据块。n 为将整个存储介质在逻辑上划分的数据区域的个数,并且从区域 1 到区

域 n 擦除次数依次减小。则

$$\left\lceil \frac{T_{max} - T_{min}}{n} \right\rceil$$

表示逻辑上相邻的两个数据区域之间在更新次数上的差值。取上界的原因是因为要避免因为数学运算不准确而出现某个数据块因为更新次数过高或者过低被排除在所有的数据区域之外。U_i 和 D_i 分别为第 i 个数据区域所包含数据扇区的更新次数的上界和下界,则这个上界和下界可以表示为:

$$U_i = T_{max} \qquad D_n = T_{min} \tag{2}$$

$$D_i = U_{i+1} = T_{min} + \left(\left\lceil \frac{T_{max} - T_{min}}{n} \right\rceil \right) \times (n - i) \tag{3}$$

或者:

$$D_i = U_{i+1} = T_{max} - \left(\left\lceil \frac{T_{max} - T_{min}}{n} \right\rceil \right) \times i \tag{4}$$

其中 U_i 和 D_i 的取值大于等于 0,i 的取值从 1 到 $n-1$,对应不同的数据区域。例如,U_2 和 D_2 分别对应第 2 个数据区域的所包括数据扇区更新次数的上界和下界,那么所有更新次数大于 D_2 小于 U_2 的扇区数据都应该被归纳到数据区域 2。

静态数据和动态数据将会在空间回收和写操作到来的时候动态地被分类到不同的数据区域中,各个数据区域的大小也会相应地进行变化。驱动程序设计者可以根据所使用存储介质的物理特性实现确定将要划分的数据区域的个数。在 Flash 存储器的使用初期,这种数据分类算法会使得有几个数据区域重叠的情况发生,这并不影响对数据的分类。随着使用时间的增加,动态负载平衡策略将会把擦除操作分配到各个数据块上,这种重叠的情况将会消失。

根据上述算法,静态数据所移动到的目的扇区应该首选那些处于数据区域 1 中的空扇区。由于静态数据被移动到了擦出频率最高的数据区域中,假定这些静态数据在未来一段时间内仍然是静态的,那么在这段时间内静态负载平衡策略不会应用在这些静态数据上,从而减少了对静态数据搬移的次数,同时降低了功耗。

4　仿真

仿真过程中使用 VHDL 语言搭建仿真平台,平台包括一个 Flash 存储器仿真模型,一个静态负载平衡单元和一个激励加载单元。构造大小为 4MB 的 RAM 来作为 Flash 存储器的仿真模型,规定其中数据块的大小为 256kB,扇区的大小为 512B,则 4MB 的 RAM 一共可以分为 16 个数据块。静态负载平衡单元用来实现触发式静态负载平衡策略,数据块分类和空间回收操作。在读写激励加载单元中加入一组读写 RAM 操作来对 Flash 存储器的仿真模型进行操作,这些激励必须保证在一段时间内对某些数据块不进行写操作,从而使这些数据块被识别成静态数据,这样才能考察静态负载平衡单元对这些静态数据所作操作的效果。同时要避免 Flash 存储器被写满的情况发生,要留出一些空闲的或者数据无效的数据块来当作静态负载平衡的目标数据块。静态负载平衡单元监测 Flash 仿真模型的状态并根据式(1)和式(4)进行静态负载平衡,数据分类和空间回收操作。

从图 4 可以看到不使用静态负载平衡策略时,静态数据永远也不会自动的向高更新率的扇区移动。图 1 中横轴为 β 的不同取值,竖轴为静态数据的搬移次数。而使用静态负载平衡策略后,可以看到,根据 β 的取值不同,Flash 存储器都会有相应的不同次数的静态数据的搬移操作。

从图 5 可以看出在使用静态负载平衡前,由于存在静态数据,所以存放静态数据的数据块擦除次数较少,而经常更新的数据块擦除次数较高。使用静态负载平衡后,由于静态数据被迫"运动"起来,所以动态负载平衡的应用范围得以更加广泛,从而使得每个可擦除数据块的擦除次数相对平均。

仿真结果还表明使用数据分类的空间回收次数比不使用数据分类时的空间回收次数少 33%,这是由于使用静态负载平衡后一次空间回收可以得到更多的空闲空间。空间回收的次数越少就意味着对数据快的擦除次数越少,这对延长 Flash 存储器的寿命有重要意义。

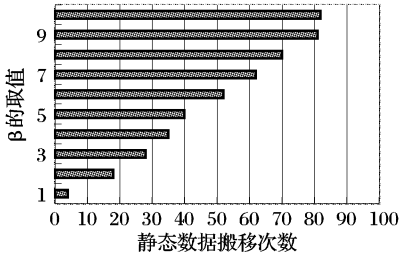


图 4　静态负载平衡策略对静态数据搬移次数的影响

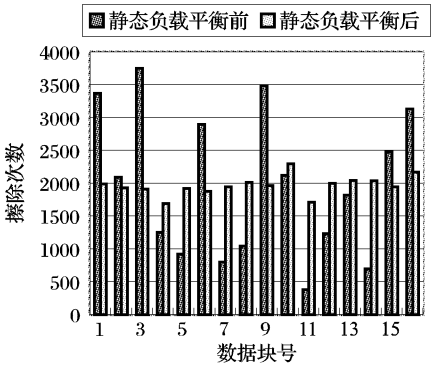


图 5　静态负载平衡策略前后每个可擦写数据块擦除次数的对比

5　结语

本文讨论了一种触发式数据分类的 Flash 存储器静态负载平衡策略,包括静态负载平衡的触发条件和一种数据区域的划分算法。通过动态地确定一个静态负载平衡的触发条件,能使静态数据在数据块间移动,从而在含有大量静态数据时也能在整个存储空间上应用动态负载平衡策略,消除了动态负载平衡的局限性。在对静态数据的搬移操作中,通过对不同属性的数据块进行动态归类进行目标数据块的选择,大大降低了静态数据搬移的次数,同时提高了空间回收的效率,降低了功耗。在当前 Flash 存储介质使用日益广泛的趋势下,可以将这种负载平衡策略应用于多种平台的 Flash 存储器的驱动程序设计中,从而大大提高 Flash 存储器的使用寿命。

参考文献:

[1] CHIANG ML, LEE PCH, CHANG RC. Managing Flash Memory in Personal Communication Device[A]. ISCE '97[C], 1997. 177 - 182.

[2] CHIANG ML, LEE PCH, CHANG RC. Flash Memory Management For LightWeight Storage System[EB/OL]. http://www.iis.sinica.edu.tw/LIB/TechReport/tr1998/tr98003.ps.gz.

[3] CHIANG ML, LEE PCH, CHANG RC. Cleaning Policies in Mobile Computers Using Flash Memory[J]. Journal of Systems and Software, 1999, 48(3): 213 - 231.

[4] KIM HJ, LEE SG. A New Flash Memory Management For Flash Storage System[A]. Proceedings of the 23rd International Computer Software and Applications Conference (COMPSAC '99)[C], 1999. 284 - 290.

[5] YUN Z. Flash Memory Technology Development[A]. Proceedings. 6th International Conference on[C], 2001.