

一种基于 NAND Flash 的 FTL 算法验证平台的设计与实现

尹子航,唐磊,沈绪榜

(西安微电子技术研究所 陕西 西安 710054)

摘 要:闪存转换层(FTL)算法的设计与开发对于固态硬盘的整体性能至关重要,由于缺乏错误检测机制,目前现有的算法验证平台检验结果与真实结果差距较大.本文提出一种基于 NAND Flash 的 FTL 算法验证平台,通过建立基于 NAND Flash 存储系统的软件平台,模拟 NAND Flash 的操作特性,增加错误检测机制,模拟真实操作环境,完成 FTL 算法的验证.本文对软件平台的整体设计和核心模块进行了分析描述.最后,在该软件平台与真实闪存设备上运行 FTL 算法完成模拟实验,实验证明两者误差范围在 7% 以内.本文提出的一种基于 NAND Flash 的 FTL 算法验证平台为验证 FTL 算法提供了一种便捷工具,同时提高了存储系统的可靠性,缩短了算法开发的周期,降低了产品成本.

关键词: NAND Flash; 闪存转换层(FTL); 算法验证平台

中图分类号: TP302

文献标识码: A

文章编号: 1000—7180(2018)11—0128—06

Design and Implementation of an FTL Algorithm Verification Platform Based on NAND Flash

YIN Zi-hang, TANG Lei, SHEN Xu-bang

(Xi'an Microelectronics Technology Institute, Xi'an 710054, China)

Abstract: At present, the design of FTL algorithm is crucial to the overall performance of the SSD. In view of the shortcomings of large gap between the verification platform and the real operation at home and abroad, this paper presents an FTL algorithm verification platform based on NAND Flash. Through the establishment of a NAND flash-based storage system software platform, simulating the operation characteristics of NAND Flash, adding the error detection mechanism, simulating the real operation environment, and completing the FTL algorithm's verification. In this paper, the overall design and core module of the software platform are analyzed and described. Finally, the FTL algorithm is run on the software platform and the real flash memory device to complete the simulation experiment. The experiment proves that the error range is within 7%. This paper proposes a NAND Flash based FTL algorithm verification platform, which provides a convenient tool to verify FTL algorithm, and improves the reliability of storage system, shortens the cycle of algorithm development, and reduces the cost of products.

Key words: NAND flash; flash translation layer (FTL); algorithm validation platform

1 引言

近年来,基于 NAND Flash 技术的固态硬盘在电子消费品市场上占有了越来越多的份额^[1]. NAND Flash 存储器作为一种低成本、非易失、单元密度高、功耗小且抗震性好的存储设备,同样受到航

空航天等安全攸关领域的青睐.

随着 NAND Flash 在固态硬盘上的广泛应用,问题也随之而来,闪存转换层(Flash translation layer,FTL)算法是固态硬盘设计的核心技术,它的设计和开发直接影响整个固态硬盘的性能^[2]. 尽管人们研究设计了各种 FTL 算法,但由于缺乏统一有

效的 FTL 算法验证平台,仍无法保证算法程序运行的可靠性.

随着 NAND Flash 速度和容量的不断提升,对管理算法的要求也越来越高,如何简便、快捷的评判一个 NAND Flash 管理算法的好坏,如何模拟 NAND Flash 的操作特性,给上层算法提供验证平台已经是迫在眉睫.

2 FTL 算法验证平台

闪存转换层(FTL)是 NAND Flash 存储器的软件中间层,它主要作用是用来消除上层对 NAND 闪存先擦后写的限制,其主要包括地址映射、垃圾回收和磨损均衡三个部分.FTL 的设计与开发对固态硬盘的研发至关重要^[3].

目前 FTL 算法验证平台成为了制约 FTL 算法研发的一个重要因素.国内外应用最多的验证方法是基于 NAND Flash 模型的仿真方法^[4].其中,国外使用较多的是美国宾夕法尼亚大学研制的仿真测试平台——FlashSim.

FlashSim 得到了较多业内人士的好评和广泛使用,但 FlashSim 在实际开发中的作用不大,更多应用于学术研究.由于其只能简单利用预设好的 trace 文件给出一个与 NAND Flash 关系不大的结果,与实际应用测试结果有时会大相径庭,不能完全模拟真实的操作环境,并且缺乏错误检测机制,无法通过预设错误(如坏块数量增多)检验算法程序的执行效率^[5].

目前国内外仍未有完善的一套通过弱化 NAND Flash 内部逻辑,针对其操作特性进行模拟,完成管理算法验证的平台.无论是国外还是国内,NAND Flash 的仿真平台处于百花齐放的状态,但其测试结果的准确性和真实性有待商榷.

针对以上情况,本文提出一种基于 NAND Flash 的 FTL 算法验证平台.通过建立基于 NAND Flash 存储系统软件平台,模拟 NAND Flash 的操作特性,增加错误检测机制,为测试管理代码提供模拟、调试的平台,更加灵活的进行算法的设计与性能评估.

平台设计的整体逻辑如图 1 所示.

该平台开发利用 C 语言编写,整体分为主机应用层、接口层和底层设备模拟层三个层面.其中主机应用层包括模块/系统测试、用例调试、代码覆盖率测试和性能测试等功能,用户通过自身需求选择不同功能利用该平台完成实验评估;接口层主要包括

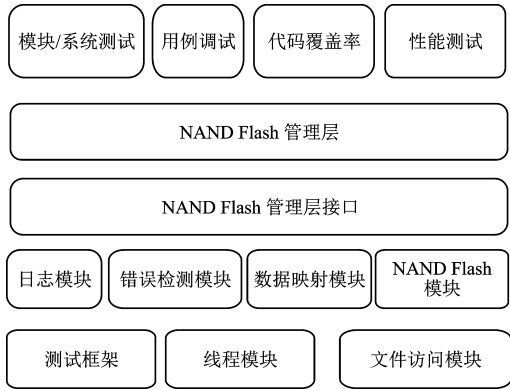


图 1 FTL 算法验证平台整体逻辑框图

NAND Flash 管理层和 NAND Flash 管理层接口,为用户操作底层设备提供可用接口,屏蔽底层逻辑,便于系统管理,同时完成对不同型号的 NAND Flash 的统一的抽象化管理,设置统一的管理接口;底层设备模拟层包括日志模块、错误检测模块、数据映射模块、NAND Flash 模块、测试框架模块、线程模块和文件访问模块.

底层设备模拟层是整体设计的核心,以文件形式构建底层闪存阵列、内存缓存和寄存器缓存,多线程模拟访问冲突,增加错误检测机制,模拟实际硬件开发中各种问题并且支持模块测试和系统测试.通过软件模拟实现 NAND Flash 存储系统,降低了开发周期,为验证 FTL 算法的可执行性提供了一种便捷平台,同时提高了存储系统的可靠性,降低了产品成本.

3 核心模块分析

3.1 测试框架模块

测试框架模块支持模块和系统测试,记录测试结果,保存压力日志并报告代码覆盖率等.

该模块主要分为模块输入、模块框架和模块输出三部分.其中模块以 YAML 格式输入测试参数和测试计划.模块输出则记录压力测试日志,并报告测试结果.模块框架则提供测试用例及支持测试用例的建立和撤销,其中测试用例在执行前建立,依次运行用例表中的条目,执行完后撤销.

YAML 输入是测试程序的起点,测试参数输入,完成 NAND Flash 的参数配置,包括块数量、块内页数、页大小及空闲(spare)区大小等,通过解析参数完成配置信息.测试配置的目的是,配置文件列表执行测试.这些配置文件中包括测试用例和测试套件,测试套件用于回归测试,测试用例用来做调试

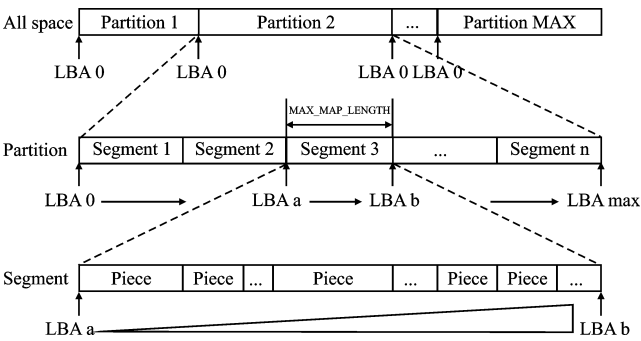


图 3 用户数据管理

于 MAX_MAP_LENGTH. 分段数量取决于分区长度. 分段通过 LBA 形式存储, 由链表管理所有分段. 链表中, 每个节点描述一个不确定的长度空间, 所有节点通过 LBA 完成排序. 每个节点, 包含我们所需信息的起始地址、数据长度、关键字和数据类型等, 这些都记录在如下结构中.

```
typedef struct UserDataItem
{
    uint32_t  uILBA;
    uint32_t  uICNT;
    uint32_t  uIType;
    uint32_t  uIKey;
} UserDataItem;
```

该模块使用一个节点描述所有空间是否已被使用, 如此相邻节点将永远不会存在空闲区.

3.4.1 数据基本操作

链表初始化, 使用前完成用户数据链表的初始化, 仅需将用户链表头(head)置为 0 即可. 如此, 第一项(item)的长度等于该段的最大长度.

新项目(item)创建, 数据写入空白 LBA, 该模块创建一个新项纪录数据并按 LBA 顺序插入到链表中.

项目删除, 链表中纪录着被写入的用户数据. 只有在新写入数据或修改数据条件下, 项目才会被删除. 这种情况下, 所有被新用户数据覆盖的项目均可被删除.

项目插入, 用户写入新数据, 我们需要插入一个新项目用来保存新数据. 插入时需要注意新项目的插入起点和结尾位置.

3.4.2 数据对比

数据对比, 主要检查从后台读出的数据是否与用户原始写入数据一致. 读取的所有数据应进行充分比较. 从数据缓存和项目中获取数据, 并进行主要

数据(地址和关键字)的比较. 如果存在任何差异, 将会反馈比较失败警报. 此外, 读取的项目在比较前应对其头尾进行特殊处理, 保证数据类型的一致.

3.4.3 数据合并

当修正数据时, 首先将要修正的项插入到链表中, 但与写入数据不同, 它将提供连续 LBA 和修正类型的项. 我们应检查和合并这些项.

为此, 我们可以在每次数据修正后检查合并项, 合并后将不再出现类似的相邻项目. 修正数据的项插入链表后, 仅存在具有连续 LBA 和相同数据的三个项目中的最大一个.

3.5 错误检测模块

错误检测模块在 NAND Flash 操作过程中进行操作失败模拟, 以此检测设计的 FTL 算法对不同错误的应对和处理效率. 该模块提供了设置错误, 检查错误和触发错误的接口.

设置错误函数由主机模块往错误表中插入错误项. 错误表是纪录错误项的表, 前面的错误项删除后续的项将往前补上. 设置错误应定义错误操作类型、错误类型、错误直、错误数据、错误比较回调函数、错误触发回调函数以及错误次数等.

检查错误函数由设备模块(device)调用, 遍历错误表找到错误项. 如果错误比较回调函数成功, 该函数将返回错误触发回调的结果.

触发错误函数是由主机定义的回调函数. 如果在设置错误时, 错误触发回调函数为 NULL, 则默认触发错误函数.

3.6 日志模块

日志模块可以在压力测试期间将命令列表记录到日志文件中, 并可以从日志文件中发送命令进行调试.

记录命令函数将设备接收到的命令纪录到文件日志中. 需强调的是, 此函数应由设备(device)替代主机调用, 原因是日志记录是设备接收到的命令. 为了正确解析命令, 可在记录命令期间使用一种固定格式的命令, 一个命令在日志文件中分配一行, 参数由空格分隔.

解析命令函数从主机模块调用日志文件中的命令完成解析, 将参数解析为可以由执行命令使用的变量.

闲置处理, 设备未收到命令的情况下将进入空闲模式, 因此主机命令和空闲操作都应记录在日志文件中. 其中, 空闲操作包括磨损均衡, 垃圾回收等. 当读取日志文件时, 设备将通过从日志文件中接受

到空闲命令来处理空闲操作,而不是让设备进入空闲模式.

3.7 NAND Flash 模块

存储在 NAND 闪存中的数据可以分成两种类型:用户数据和元数据. 上层模块将解析主机命令,根据数据类型执行不同操作.

用户数据以扇区 (sector) 为单位,是主机写入和读取的数据. 元数据则是 FTL 管理 NAND 闪存,需要记录到 NAND 闪存中的数据,例如初始化坏块信息,映射信息等^[6].

该模块包含三个子模块和两个数据流. 三个子模块分别是 NAND 管理模块、用户数据管理模块和元数据管理模块. 两个数据流分别是用户数据流和元数据流.

NAND 管理模块管理闪存信息,例如页面大小,坏块数量等. 其主要功能是加载 NAND 闪存信息,包括芯片 ID、坏块表和页映射表等;覆盖测试,启用或禁用覆盖检测;块管理,坏块管理和块擦除等^[7].

用户数据管理和元数据管理模块用于独立处理用户数据和元数据.

元数据管理,由于所有元数据都有用,所以数据应逐字存储. 为避免文件过大导致降低操作效率,每个块都由一个独立的文件描述. 所有这些块由块链表管理,每个块都是其中一个节点.

用户数据管理模块中,若页面的用户数据量很小,可将一个芯片中所有用户数据存储到一个文件中. 由于 SSD 实际空间相当大,处理所有数据将花费太多时间和空间. FTL 有必要管理用户数据,并通过一些技巧减少用户数据量^[8].

ROR1. 类型 2 是映射到旧数据 (old data),就如不适时的更新造成的 ERROR2.

为了检测错误,用户数据需携带额外数据. 包括 LBA 信息 (LBA info), 4 字节大小,单位为扇区,用于检测类型 1 的错误;写标志 (Write Tag), 4 字节大小,判断数据是否为最新写入的,用于检测类型 2 的错误;闪存空闲区信息 (Spare info), 本文选取 16 字节空闲区大小,用来保存扇区状态信息.

3.8 其他模块

主机应用层,包括各模块/系统测试、用例 (case) 调试、代码覆盖率测试和性能测试,针对用户不同测试目的选择相应模块.

接口层主要包括 NAND Flash 管理层和 NAND Flash 管理层接口,管理不同型号 NAND Flash,为用户操作底层设备提供统一接口,主要是对 NAND Flash 的基本操作如读取、写入和擦除等.

4 平台实现与测试

该算法验证平台开发基于 code-blocks 集成开发环境,运行在 Linux 操作系统. 选择镁光 MT29F64G08CBAAA 模拟底层 NAND Flash,该闪存颗粒划分为 4 096 个块,每个块含 256 个页,每个页大小为 (8 192 + 448) 字节,其中后 448 字节是 spare 空闲区存放命令后的状态信息,我们选取前 16 字节进行模拟. 该型号颗粒的页写入时间为 1 300 μ s,页随机读取时间最大为 50 μ s,块擦除时间为 3ms. 软件平台设计框架如图 5.

为测试平台对 FTL 算法的验证效果,我们将 DFTL 算法置入该验证平台中,进行仿真实验^[9]. 已知 DFTL (demand-based FTL) 是一种根据负载访问特点动态加载映射项到缓存中的知名 FTL 算法,同时 DFTL 也是 Flashsim 自带的几个 FTL 算法之一^[10]. 通过编写测试用例进行 FTL 算法逻辑和性能测试并与真实闪存设备的模拟测试结果对比,误差范围在 7% 以内.

5 结束语

本文通过分析固态硬盘 (SSD) 开发环节的局限性,针对目前国内外仿真平台的缺陷,提出一种基于 NAND Flash 的 FTL 算法验证平台. 该平台通过建立基于 NAND Flash 存储系统软件平台,模拟 NAND Flash 的操作特性,增加错误检测机制,为测试管理代码提供模拟、调试的平台. 该平台已完成软件开发和初步测试,实验证明该平台能有效降低开

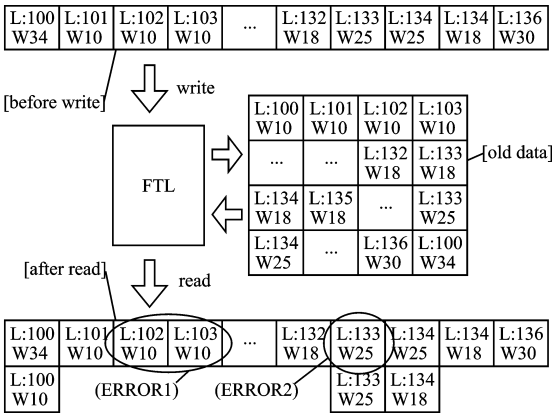


图 4 FTL 地址映射管理

如图 4,FTL 发生错误分成两种类型,类型 1 是 LBA 映射到错误的地方,如逻辑错误造成的 ER-

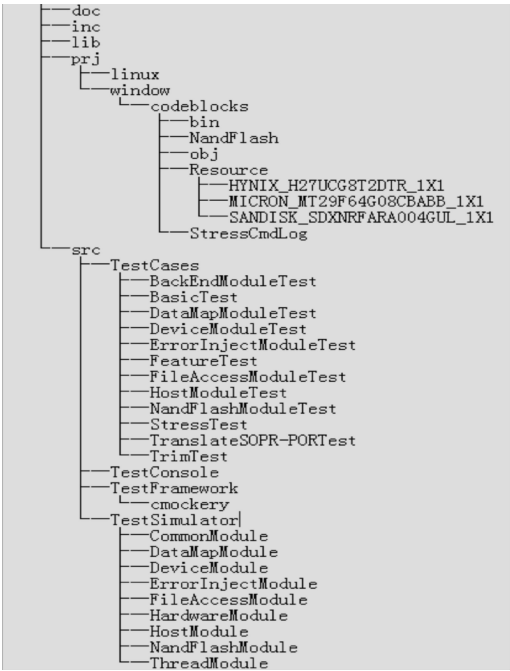


图 5 软件平台设计框图

发周期和产品成本,为验证 FTL 算法的可执行性提供了一种便捷工具,同时提高了存储系统的可靠性.

参考文献:

[1] Park C, Cheon W, Kang J U K, et al. A reconfigurable FTL (flash translation layer) architecture for NAND flash-based applications. [J]. ACM Transaction on Embedded Computing Systems, 2008, 7(4): 1-23.

[2] 郑文静,李明强,舒继武. Flash 存储技术[J]. 计算机研究与发展, 2010, 47(4): 716-726.

[3] 蔡晓乐,张亚棣,李亚晖,等. 基于区域页级映射的闪存转换层设计[J]. 微电子学与计算机, 2016, 33(4): 32-36.

[4] Kwon S J, Ranjitkar A, Ko Y B, et al. FTL algorithms for NAND-type flash memories. [J]. Design Automation for Embedded Systems, 2011, 15(3-4): 191-224.

[5] 张雯,崔建杰,张新. 一种多通道 NAND Flash 阵列的坏块管理方案[J]. 电子器件, 2014(5): 816-821.

[6] Lee Y S, Barolli L, Lim S H. Mapping granularity and performance tradeoffs for solid state drive[J]. Journal of Supercomputing, 2013, 65(2): 507-523.

[7] Lu K, Jin P Q, Yang P Y, et al. Adaptive in-page logging for flash- memory storage systems[J]. Frontiers of Computer Science, 2014, 8(1): 131-144.

[8] 时正,陈香兰,纪金松,等. 大容量 NAND Flash 文件系统中的地址映射算法研究[J]. 小型微型计算机系统, 2010, 31(1): 155-159.

[9] Seungjae Baek, Heekwon Park, Jongmoo Choi. Improving the Reliability and Performance of the YAFFS Flash File System. [J]. IEICE Trans. inf. & Syst, 2011, 94(12): 2528-2532.

[10] Park C, Cheon W, Lee Y, et al. A Re-configurable FTL (Flash Translation Layer) Architecture for NAND Flash based Applications[J]. Acm Transactions on Embedded Computing Systems, 2008, 7(4): 202-208.

作者简介:

尹子航 男, (1994-), 硕士研究生. 研究方向为计算机科学与技术. E-mail: 913—64066@qq. com.

唐 磊 男, (1973-), 博士生导师, 研究员. 研究方向为嵌入式系统设计.

沈绪榜 男, (1933-), 中科院院士, 研究员, 博士生导师. 研究方向为嵌入式系统设计.