

税控收款机 Flash 文件系统的设计与实现

林志洁¹, 吴吉义²

(1. 浙江科技学院 信息与电子工程学院, 杭州 310023; 2. 浙江广播电视大学 远程开放实验室, 杭州 310012)

摘要: 针对税控收款机设计了一种 Flash 文件系统, 通过 Flash 空间分配实现方式优化、断电保护和空间回收等关键技术, 并结合嵌入式数据库设计思想, 向用户提供方便可靠的数据导入导出及日常检索功能, 从而为税控收款机的推广提供了系统软件支持。

关键词: 税控收款机; Flash 存储器; 嵌入式系统; 文件系统

中图分类号: TP311

文献标识码: A

文章编号: 1671-8798(2007)01-0018-05

Design and Implementation of Flash File System in Tax-Controlled Cash Register

LIN Zhi-jie¹, WU Ji-yi²

(1. School of Information and Electronic Engineering, Zhejiang University of Science and Technology, Hangzhou 310023, China;
2. Remote Open Laboratory, Zhejiang Radio & TV University, Hangzhou 310012, China)

Abstract: A kind of embedded Flash file system is designed especially for tax-controlled cash register. By optimizing structure arrangement scheme on Flash, the key techniques, such as realizing defragmentation, keeping Flash file system operating correctly with unexpected interruption and integrating with ideology in database design, can be realized. The file system affords convenient functions such as querying, inputting and outputting data safely. It provides system-level software supporting for generalizing tax-controlled cash register.

Key words: tax-controlled cash register; flash memory; embedded systems; file system

在税源紧缩形势下, 国家税务总局出台了新税控收款机(以下简称税控机)标准。税控机是具有税控功能的电子收款机, 它需要保证经营数据的正确生成、可靠存储和安全传递, 并可实现税务机关管理和数据核查。Flash 存储器是税控机存储程序和用户数据的主要介质, 但 Flash 存储器本身具有一些独特的性质, 加上税控机对存储文件的要求, 与之对

应的文件系统必须具有文件管理、断电保护、数据检索等功能。本文将结合巨鹰集团的税控机产品开发项目, 介绍该类文件系统的设计。

1 Flash 存储器的工作原理

Flash 存储器具有以下特性^[1]:

1) 数据只能写到 1 位上, 而不能写到 0 位上;

收稿日期: 2006-12-12

基金项目: 浙江科技学院启动基金项目(200504)

作者简介: 林志洁(1980—), 男, 浙江平阳人, 助教, 硕士, 主要从事嵌入式系统软件及计算机网络的研究。

2) 为将 0 变为 1, 要对存储区进行擦除操作;

3) 擦除以块为单位串行进行;

4) 每个块的可擦除次数为 10 万~100 万次。

因此, Flash 存储器在读取方面与普通的 SRAM 存储器类似, 但在写操作上有很大的区别, 包括以下两种情况^[2]:

1) 写新文件: 在新申请的空间中“写入”对应数据即可。

2) 修改文件: 不能在原文件上修改, 必须申请一块新空间存放修改后的数据, 再将原文件设为无效。

2 税控机文件系统的特点

税控收款机中存放的主要文件包括税控系统软件、汉字库、用户数据。其中, 前两项一般较大, 机器初始化时存入 Flash 中, 只有在系统版本更新时才可能被修改。用户数据包括操作人员信息、商品信息、发票明细记录、日交易记录、申报记录、每卷汇总记录等。这些文件由用户在使用过程中逐步存入 Flash 中, 但也不大会被修改(商品变动、退货、退发票时可能会修改); 然而, 这类数据要经常被查询检索, 如制作报表和税务稽查^[3]。因此, 有必要考虑对 Flash 上的数据的检索性能。

另一方面, 税控机运行环境不稳定, 很容易在断电等突发事件中发生错误甚至丢失数据, 文件系统的写入、擦除等操作都对系统的异常中断非常敏感, 于是, 为了保证数据的可靠性, 必须在文件系统中建立一定的纠错和保护机制。

3 税控 Flash 文件系统的设计

本文将为税控机设计一种方便、安全的文件系统, 其方案和实现细节如下。

3.1 文件系统的结构

在功能上文件系统可以分为 3 层: 用户接口层、文件管理层和 Flash 设备抽象层, 如图 1 所示。

用户接口层对整个文件系统进行封装, 结合数据库设计思想, 屏蔽下层实现细节, 为上层税控应用提供统一的、标准的 API 接口, 把用户对文件的操作请求提交给文件管理层来处理。

文件管理层保证用户可以安全地将数据存储于 Flash 上, 即使意外断电也不会导致数据丢失, 把用户接口层对文件的操作转换为物理逻辑上的操作, 并提交给 Flash 驱动层处理, 从而对上层用户而言, 屏蔽了 Flash 设备特殊的写方式。在该层实现大部

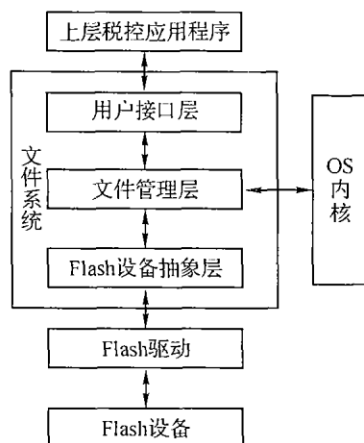


图1 体系结构示意图

分的文件系统功能、Flash 文件系统特有的功能, 例如所有的文件级操作以及断电保护、垃圾回收、日志记录等, 都在该层实现。

Flash 设备抽象层提供对 Flash 存储设备驱动的综合管理机制, 它不仅方便上层对这些 Flash 设备的访问, 而且方便了建立在这 Flash 设备上的上层核心模块的设计。该设计使得本文件系统可以方便地在不同的 Flash 上实现。

其中, 用户接口层和文件管理层是本文件系统的特色与核心, 它的实现比较复杂, 本文仅对其重要特色内容做介绍。

3.2 用户接口层

税控应用程序通过 API 获取用户接口层的功能支持。该层提供了类似数据库操作的一般功能, 包括基本表的创建和删除, 索引表的创建和删除, 向基本表中添加记录值, 打开关闭基本表, 查找记录或属性列值, 按某属性列值的条件删除或修改记录, 以及其他一些辅助性的 API 调用。

为了实现本层的功能, 设计了一些共享数据结构, 原理如下。

3.2.1 基本表的控制节点结构 系统初始化时, 根据宏定义的系统可同时打开的基本表数目, 分配基本表的控制节点结构 BasicTbCtrlNode 结构空间。在打开或创建基本表的时候, 分配该结构, 并对该结构初始化赋值, 包括索引和缓冲区相关的变量; 如果是创建, 只对部分变量赋值, 其他变量在具体操作中进行。在关闭基本表的时候从基本表收回该结构, 系统运行结束的时候释放结构空间。

3.2.2 索引表 索引的组织形式比较多, 本系统中不支持多级索引, 因此, 根据实际情况, 经分析本文

从排序文件上的简单索引、辅助索引、B 树、散列表等单级索引表组织形式中,选择了排序文件上的简单索引。由索引控制结构头和索引数据项组成。

3.2.3 输出缓冲区 输出缓冲区按循环双向循环队列方式进行组织管理。如果队列满,按照 FIFO 原则从 front 头指针出队列。出队列的数据并不一定是当前要输出的记录,当前要输出的记录用 current 指针指示。

3.2.4 基本表 用户数据在基本表中的二进制文件格式见图 2。

表头控制信息
列 0 的定义
列 1 的定义
.....
列 $n-1$ 的定义
记录行 0 数据
记录行 1 数据
表数据区

图 2 基本表的存储格式

第一部分是基本表控制结构,其中定义了该基本表的偏移量信息 and 对该基本表的操作信息;第二部分是列定义,记录了基本表各个属性列的定义信息;第三部分是基本表的数据区,该部分存放了基本表的所有记录数据。

创建基本表时,用户输入基本表文件名和属性列定义等信息后,系统为该基本表分配一个基本表控制点结构。将基本表的属性列定义和控制结构信息放在无格式的表内容前面,作为文件内容参数,调用文件管理层 API,创建一个以基本表名命名的文件。创建基本表成功后,对该基本表输出缓冲区的三个指针进行初始化。

3.3 文件管理层

3.3.1 Flash 上的结构布局 在 Flash 文件系统的设计中,大多考虑的是读写操作相当的应用环境,所以都把精力放在断电保护和均衡磨损上。而在本系统的应用环境中,写操作很少,以读操作为主。基于此,综合比较几种文件空间管理技术,例如连续分配方式、索引连接表分配方式、 i 节点的分配方式,采用了连续分配方式。这种分配方式简单、读写性能好而且维护方便^[4],对于税控机应用已经足够。针对 Flash 的擦写特性和应用对读性能的要求,本文考虑把文件名、文件创建时间、文件开始位置、文件长度等不轻易改变的文件头结构信息存放在一起,

形成文件的主要目录结构。为了解决断电安全问题和空间回收的实现,在对 Flash 使用寿命做的详细分析后,在 Flash 上设计了一个空闲块替代传统的 RAM 中转设计,另外设计了两个数据结构存放断电恢复和空间回收的参考信息和跟踪信息。如上所述,本文件系统在 Flash 上的结构布局如图 3 所示。

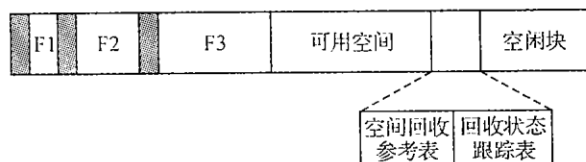


图 3 Flash 空间结构布局

图中阴影部分代表文件头结构,F1、F2、F3 是 3 个文件体,可用空间是尚未分配的空间。

1)文件头:图中的阴影部分。这是每个文件必须的开销。

2)空间回收参考表:在进行空间回收时,对每个有效(非删除)的文件在这个表里创建一个表项。

3)回收状态跟踪表:在进行空间回收时,记录空间回收过程所处的状态。Flash 数据区上一共有几个块,这个表就有几个表项。每个表项长 4 个字节,代表一个块当前所处的状态,回收进程通过修改各个表项的位状态来记录当前状态。

4)空闲块:取代普通文件系统 RAM 写缓存,用来在空间回收过程中备份当前整理块数据,使得数据在意外断电后不会丢失。该块必须大于任何其他 Flash 块,由一个以上 Flash 块组成。

3.3.2 优化的文件修改 一般情况下,当一个文件被修改以后,源文件会被标识为废除,然后在 Flash 尾部分配一段新的空间来存放该文件的新版本^[6]。如果只是做了一些微不足道的修改的话,这样做的代价显得十分巨大。因此,作为一种优化的方法,笔者设计了一种“原处修改”型的文件。这种文件在创建的时候将被额外多分配一定的空白空间,这样,对该类文件做小量修改的时候,就可以不用经历擦除-重写这个步骤。如果对该文件的修改不大,则该操作将直接写在该文件尾部可用空间里。

系统运行过程中,文件可以以只读/修改/创建模式打开。文件类型和打开方式通过标识文件头结构中的 flag 实现。

3.3.3 断电保护 除在文献[5]中提到的硬件电容作为税控机意外断电的保护措施外,还需软件的配合。软件的断电保护的简要实现过程如图 4 所示。

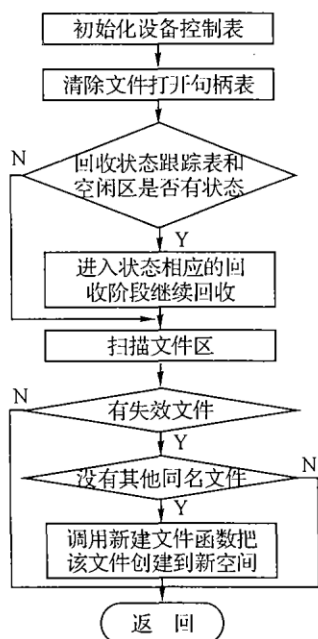


图 4 断电保护恢复过程

其中涉及的函数描述如下:

`fsdevtblinit()`:把设备表中的 Flash 设备信息拷贝过来初始化设备控制表 `fsDeviceTbl[]`。

`fsclear(TDEV * tdp)`:清除 `fileslot` 表,表示没有文件处于打开状态;如果传进来的 `TDEV` 指针是空的,那么可以认为是全局初始化,于是继续清除任何信息。

`fsfixup(int verbose)`:根据空闲块内容和数据区末端中的回收进程状态表,判断空间回收进程是否曾经在运行过程中被意外中断,如果没有被意外中断过,则就此返回;否则,根据状态表中的状态和空闲块中的内容,定位断电前系统所处状态,调用 `fsclean()` 运行回收进程直至成功结束。

`fsstalecheck()`:该函数用来检查是否有文件在修改过程中,没来得及写完新文件就被意外中断了。对于检查的结果可能有 3 种情况,现将这 3 种情况及其处理方式简单描述如下:

- 1) 没有被标识为失效的文件,直接返回。
- 2) 有一个标识为失效的文件但没有其他文件和他同名,这种情况对应于在修改文件过程中旧文件已经被标识为失效而新文件还没来得及写入或没完整写入的情况。对这种情况,因为文件标识位只能从有效改成无效,即从 1 擦成 0,而不能再改回 1。所以,要把该文件拷贝到新空间中,标识为有效文件,才能把旧文件删除。
- 3) 有一个失效文件和一个同名的有效文件。这

种情况表示该文件因为修改成功被删除,故可直接返回。

3.3.4 空间回收 选择什么技术来解决这个问题绝大部分取决于该系统所将应用的环境、可获得的硬件资源以及对功能和复杂度这两者的取舍。本文件系统的设计目标是保持文件简单线性地存储,以保证应用程序可以认为文件在 Flash 上是顺序存储的,因此,采用比较简单的方法。

本文采用的空间回收的方法是不用 RAM 空间,而是利用一块称之为空闲块的 Flash 空间作为中转。这个块的 Flash 空间是提前就分配好的,它被布置在 Flash 的最后一块上,目的是不会影响应用程序对数据的顺序访问。除此之外,还需要在文件系统的结尾处保留足够的空间,为每个有效文件存放一个空间回收参考记录,以及跟踪回收进程状态用的 128 个字节的回收状态跟踪记录。在空间回收过程中,保存足够的状态以保证断电后从这些状态恢复。这些状态被保存在 Flash 上的跟踪状态表上,即使断电也不会丢失。简要回收过程如图 5 所示。

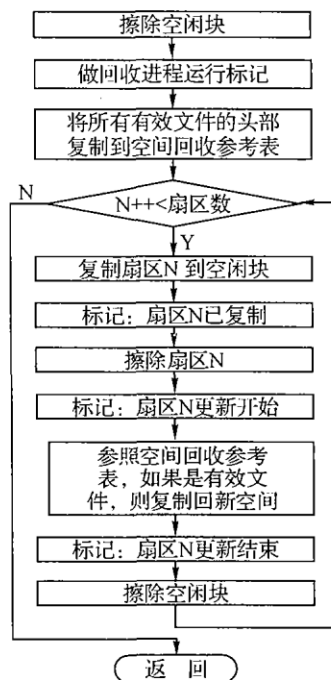


图 5 空间回收过程

在空间回收过程中,回收进程的不同阶段每一个对 Flash 的操作都对应一个状态,这样,在系统突然断电后,可以通过这些状态来判定断电前系统正在做的操作,从而达到断电保护的目的。这样设计的好处是即使在空间回收的过程中断电也不会丢失数据了;缺点是这块作为中转空间的 Flash 将被频

繁擦写,它将最快到达 Flash 的擦除次数上限,从而缩短了整个 Flash 的寿命。这个上限将在什么时候到达取决于 Flash 上分配了多少个块和 Flash 上的数据删除重建的频率。

3.3.5 Flash 寿命的考虑 目前的 Flash 制造技术对应的 Flash 产品的块擦除上限为 10 万~100 万次。假设 Flash 文件系统管理 21 个块的 Flash 空间,另外 1 个块 Flash 空间作为空闲块,并假设最坏的空间回收情况(所有的 21 块都受影响),那么在一个每块有 10 万次擦除上限的 Flash 上每天运行一次空间回收进程,空闲块损坏之前仍然有 13 年的寿命($100\,000/21/365=13$)。在最新的税控收款机国家标准中,对存储器的要求是:存储至少 5 年的日交易税控数据。因此,在大部分的 Flash 将支持每个块 100 万次擦除上限的特性趋势下,本文没有必要设计“均衡擦写”,因为那将增加底层实现复杂度。

4 测试

测试分为功能测试和性能测试两部分。功能测试一切正常,本文不细述。下面是性能测试的情况。

4.1 文件系统资源消耗

Flash 文件系统的资源消耗,主要包括程序代码开销、处理器占用时间、运行时内存开销以及文件系统的 Flash 存储器消耗。结果如表 1 所示。

表 1 资源消耗测试结果

实验条件 Flash 存储器规格:1kB/扇区×2 000 扇区,预先存储 20 个文件,打开其中 5 个文件				
实验结果	内存开销	程序代码开销	7.24 k/2 M=0.36%	
		文件系统 Flash 消耗	12.6 k/2 M=0.63%	
		输出缓冲区	0.5 k	
		文件打开跟踪表	0.45 k	1.28 k/8 M=
		基本表控制表	0.25 k	0.016%
		其他	0.08 k	

4.2 可靠性

嵌入式系统中的 Flash 存储器主要面临两大不稳定因素:一是 Flash 存储器本身可能出现物理性的损坏;二是嵌入式系统面对较多的突发断电或重启,造成 Flash 存储器写操作的异常终止。

关于物理性损坏问题,已经在前面章节做过分析,不再做测试。笔者进行了断电重启测试,实验结

果如表 2 所示,从中可以看到,即使在写操作异常频发的恶劣工作条件下,本文件系统也可以保持可靠工作。

表 2 可靠性测试结果

实验条件	Flash 存储器规格:1 kB/扇区×2 000 扇区,预先存储 20 个文件文件长度分别为 $k \times 100$ kB($k \leq 5$),模拟各种写操作进行当中,发生系统断电类事故,造成写操作异常中止。	
实验结果	实验次数/次	100
	导致系统重启后报错次数/次	80
	REFS 启动后自动修复次数/次	80
	Flash 上已经存储的文件受损情况	0 文件/0 次

5 结 语

本文针对税控机的特殊应用,提出了一种 Flash 存储器空间管理方案,在用户接口层引入简单的数据库思想,满足了税控机上层软件的检索需求;利用 Flash 末端的空闲区和状态跟踪表的设计,解决了 Flash 文件系统的断电保护问题,达到了提高税控收款机运行效率、方便上层应用软件开发和移植的效果,并在税控机产品巨鹰 GE3000 中得到了应用。

参考文献:

- [1] WOODHOUSE D. JFFS: The Journaling Flash File System[DB/OL]. (2006-10-29). <http://sources.redhat.com/jffs2/jffs2.pdf>.
- [2] DAI H, NEUFELD M, HAN R. ELF: an efficient log-structured flash file system for micro sensor nodes [C]//Proceedings of the 2nd international conference on Embedded networked sensor systems. New York, USA: ACM Press, 2004.
- [3] GB 18240.1—2003, 税控收款机[S].
- [4] 陈智育. 嵌入式系统中的 Flash 文件系统[J]. 单片机与嵌入式系统应用, 2002(2):5-8.
- [5] 魏建苗,李莹. 基于 ARM 微处理器和 μ Clinux 的税控收款机的设计与实现[J]. 工业控制计算机, 2005(1): 56-57,59.
- [6] 郑良辰,孙玉芳. 日志文件系统在嵌入式存储设备上的实现[J]. 计算机科学, 2002(1): 72-74,52.