文章编号:1671-4598(2019)02-0212-04 DOI:10.16526/j.cnki.11—4762/tp.2019.02.046 中图分类号:TP274.2 文献标识码:A

嵌入式系统大容量 NAND Flash 存储器分区管理设计

李远哲[1]，贺海文[1]，万 丽[1]，李 妍[2]，赵 峰[3]
(1. 中国人民解放军 32184 部队，北京 100072；2. 北京北方车辆集团有限公司，北京 100072；
3. 北京航天测控技术有限公司，北京 100041)

摘要：针对嵌入式系统中对大容量 Flash 存储器数据存储管理的需求，对大容量 NAND Flash 存储器分区、数据写入机制进行研究，借鉴 FAT 文件管理系统的理念，通过定义和动态维护分区属性数据结构的方法，建立了一种简单易行的分区管理机制，控制数据存储管理，并给出了具体的数据结构和设计流程图；结合实际工程应用进行了实验验证和分析，应用结果表明，该方法软件设计简单，资源需求低，实现了对大容量 NAND Flash 存储器的分区化管理，方便了用户对目标数据的快速检索定位，提高了数据使用效率，同时保证了数据存储的完整性、实时性、正确性，并大大降低了对上位机数据处理的难度；该方法的提出，为嵌入式系统中大容量 Flash 存储器的使用管理提供了新的思路，具有较高的实用性和推广应用价值。

关键词：NAND Flash；嵌入式系统；存储器分区设计

Design of High—Capacity NAND Flash Memory Partition Management in Embedded System

Li Yuanzhe[1], He Haiwen[1], Wan Li[1], Li Yan[2], Zhao Feng[3]
(1. 32184 troops of PLA, Beijing 100072, China; 2. Beijing North Vehicle Group Co., Ltd., Beijing 100072, China;
3. Beijing Aerospace Measurement and Control Technology Co., Ltd., Beijing 100041, China)

Abstract: In view of the data recording and management requirement of high—capacity flash memory for embedded system, a innovate mechanism of partition management has been established by defining and dynamically maintaining partition attribute structure based on the research high capacity NAND flash memory partition, data writing mechanism and the ideas of FAT file management system. The data structure and design flow chart are presented to describe it. To combine with the experiments of actual engineering application, the results show that it can be used to realize the partitioning management of high—capacity flash memory data recording, and rapid detect the target data. It can enhance the efficiency of data and simultaneously keep the dada with integrity, good real—time feature and accuracy, which reduces the difficulty of data processing, Hence, it is significant to provide a new method on the use of high—capacity flash memory with higher practical and popular value.

Keywords: NAND flash; embedded system; memory partition design

0 引言

在数据采集、状态监测以及故障诊断等应用领域，随着目标对象的技术含量越来越高、结构组成越来越复杂，所需要的技术参数越来越多，对嵌入式系统存储容量的要求也越来越高。大容量 NAND Flash 存储器的出现和应用，意味着可以存储更大量的数据或者说更长的数据覆盖周期，为更深层次的数据挖掘奠定了数据基础。

通常，对大容量 NAND Flash 存储器的使用主要采用直接使用和移植 FATS 文件系统两种方法。直接使用法是最基本和直接的方法，每次的下载或删除总是从存储器零地址开始，无法控制所需数据的大小，同时，删除后再次写入也是再次从原起始地址开始，造成存储器的不均衡使用，影响使用寿命；移植 FATS 文件系统法是在嵌入式系统中软件移植了 FATS 文件系统，其优点不用赘述，但这种方法对硬件性能和资源的要求相对较高，而且软件设计相对复杂，因此在应用于诸如成本限制严格、硬件系统资源相对不足的场景中时，为实现分区管理而占用过多的系统资源或为此而需要提高嵌入式系统性能不是一个最佳选择。

实际经验表明，离线分析中所需要的关键数据通常只是全部存储数据的某一个或几个片段，并不需要把所有的数据全部下载。因此设计一种分区管理策略，尽量减少数据下载数量，使用户可以根据故障时机等相关信息仅下载关键数据，同时，实现存储器存储区间的均衡使用，是提高嵌入式系统大容量存储器使用性价比的最佳选择。

鉴于上述分析，本文提出通过建立 NAND Flash 存储器分区属性的方法，描述数据分区，控制数据存储，形成

收稿日期:2018-07-31； **修回日期**:2018-08-27。

作者简介:李远哲(1973-)，男，河南封丘人，硕士研究生，高级工程师，主要从事仪器开发和测试技术、车辆试验与鉴定、检测技术与自动化装置方向的研究。

多个数据文件。用户通过分区属性获得数据文件中的数据起始时间、结束时间以及进行数据写入、导出、删除等操作所需要的相关信息，使用户对数据文件的操作类似于常规文件管理系统下对标准文件的操作。该方法无需升级硬件系统、无须移植文件管理系统、无须编写复杂的驱动模块，在非易失性铁电存储器（FRAM，如 FM25640－S）的辅助下，实现了分区管理，解决了对大容量 NAND Flash 存储器的分区管理问题。

1　NAND Flash 分区设计思想

1.1　总体设计思想

NAND Flash 分区设计的总体思想是将其全部可用空间划分为多个由指定个数的连续块（Block）构成的分区，每个分区作为一个独立的“文件”来看待，并以文件属性的方式表征分区属性，描述分区在 NAND Flash 存储器中的位置、相关操作状态以及该属性在 FRAM 中的存储地址等。套用我们熟悉的硬盘及 FAT 文件系统的概念，本思想可以形象地描述为将一个大硬盘划分为多个具有唯一编号的微硬盘，每个微硬盘上只存储一个文件名固定的文件。这里文件名以编号的形式出现，文件属性被定义为除指明文件状态、写位置等属性外，还包含微硬盘起始地址、空间大小等一些有别于标准文件属性的属性，即包含了分区和分区内数据的属性。分区属性包括分区标号（文件号）、起始地址以及文件属性在 FRAM 中的存储位置等；分区内数据的属性对应文件属性中动态变化的部分，包括数据结束日期时间、写位置等。文件属性中的分区属性在 Flash 格式化时设定，分区内数据的属性随文件的大小动态维护。

上述设计思想的数据结构描述如下：

```
define MAX _ FLASH _ FILE _ NUM 20 //定义分区总数
define MAX _ FILE _ LENGTH _ PAGE 12800//分区空间（文件）大小，以 Flash 页为单位
typedef struct
{
byte unFOM; //文件标志：bit0－空/bit1－满/bit2－正在写/bit3－被导出/bit4－被覆盖
byte bFileNo; //文件号
byte bPhyFileNo; //物理文件号，决定文件属性在 FRAM 中的实际地址
unsigned long ulStartPage; //文件起始 Flash 页
DWORDTIME stCreateDate; //数据起始日期时间
DWORDTIME stEndDate; //数据结束日期时间
unsigned long ulPageWtr; //当前写位置
} FLASH _ FILE _ PREP;
```

FLASH_FILE_PREP 结构所定义的文件属性保存在 FRAM 中，并在数据操作过程中被动态维护。其中，unFOM 定义了当前文件的操作状态，包括空、满、正在写、被导出、被覆盖等，对应位置 1 表示有效，0 表示无效；bFileNo 定义了当前文件的文件号，对应 Flash 中的一个唯一分区，是该分区的识别标识；bPhyFileNo 这里称为物理文件号，它实际表明了对应文件的文件属性保存在 FRAM 中的实际地址序号，在操作中可通过该编号获得文件属性在 FRAM 中的实际保存地址；stCreateDate 和 stEndDate 分别表示了保存在该文件中的数据的起始时间和结束时间，表示为“年、月、日、时、分、秒”，为索引定位数据提供时间标识；ulStartPage 指定了该文件在 Flash 中的起始页地址，也可以形象地称为是“文件首地址”；ulPageWtr 表示当前写位置，是相对于文件首地址的偏移量（偏移页数），以用于计算当前数据将要写入到 NAND Flash 存储器哪一页，同时该元素也表征了文件中有效数据的大小。

上述代码中做了两个宏定义，MAX_FLASH_FILE_NUM 表示整个 Flash 存储器被划分为多少个分区，即文件个数；MAX_FILE_LENGTH_PAGE 表示每个分区分配的总页数，即文件最大容量。这两个宏定义由用户根据实际工程的需要，结合所选用的 Flash 存储器指标和具体存储管理需求设定。

在首次（或后续必要时）使用时，需要对 Flash 存储器进行格式化，即完成分区和分区内数据属性的初始化，其操作过程描述如下：

1）分配文件号，并初始化文件属性，除文件空状态位置 1 外，其它属性位均为 0；

2）分配物理文件号，确定当前文件属性在 FRAM 中的存储地址序号（＝文件号）；

3）设置文件的起始页码（＝文件号 × MAX_FILE_LENGTH_PAGE）；

4）文件写位置初始化（＝文件的起始页码）。

4）保存文件属性至 FRAM。

5）选定 0 号文件作为当前文件，并设置数据起始日期时间。

上述结构中用到了自定义的数据类型 DWORDTIME，用于表示日期时间，其定义如下：

```
typedef struct
{
byte year:5;//年
byte mon:5;//月
byte sec:6;//秒
byte day:5;//日
byte hour:5;//小时
byte min:6;//分钟
} DWORDTIME ;
```

为实现上述设计思想，FRAM 需要存储每个文件的文件属性，并在 NAND Flash 的读写等操作过程中实时维护。由于文件属性直接决定了整个 NAND Flash 存储器以及当前文件的使用情况，其正确性与否直接影响数据的读写位置以及相关操作，所以保证文件属性的正确性具有极其关键和重要的作用。由于嵌入式系统本身的特性，任何情况下的掉电都是“正常”的，所以必须考虑采取措施保证对文件属性保存的正确性。

对于使用 FRAM 的嵌入式系统来讲，影响文件属性正确性的因素主要包括 FRAM 的可靠性程度和软件策略的设

计。对于 FRAM 的可靠性程度可以由硬件本身来保证。对于软件策略来讲，主要是对数据保存策略和完整性判读的处理，这里采用了“三取二”的保存与判读策略，即对要保存的数据在不同的存储空间连续保存三次，取其中两次相一致的数据作为最终结果。其原理是通过软件设计强制这种存储操作是连续的，若第 i 次存储未完成，则另外两次的存储将不会进行，也就是说，每次的存储操作仅针对当前存储区，其他的两个存储区均不会被更新，因此，必然存在两次完全一致的存储结果，而且这两次完全一致的存储结果必然是完整且有效的。

根据用户对整个 NAND Flash 空间的分区定义，将对 FRAM 划分为 MAX _ FLASH _ FILE _ NUM 个连续存储空间，每个空间的大小为 3×sizeof（FLASH _ FILE _ PREP）字节，这样，每个文件号对应的 FRAM 存储起始地址的为：bPhyFileNo×3×sizeof（FLASH _ FILE _ PREP）。由此，根据“三取二”的判读策略，无论在任何情况下，在连续保存的 3 次文件属性中，总存在至少两次是相同的，取这两次结果中的一个作为最终结果使用

在对 NAND Flash 存储器进行格式化时，一方面分配了文件号，同时也确定了物理文件号，分配了每个文件的文件属性在 FRAM 中的存储地址和空间，初始化了文件属性中的相关参数，而且，在格式化时，对于每个文件的文件属性都进行了 3 次存储，为后续的利用奠定了基础。

1.2 数据写入的实现

在系统上电后，若是首次操作，则必须进行格式化操作，以获得分区和分区内数据属性的初始参数，并选定首个操作文件；否则，则从 FRAM 中读取当前操作的文件号，计算对应的文件属性在 FRAM 中的地址以读取相应的文件属性，至此存储器即可进入正常的工作状态。根据 NAND Flash 存储器的操作特性，数据的读写是按页进行，因此需要在缓存中的数据达到一页的容量时，先将数据写入 NAND Flash 存储器，即写入当前文件的 ulPageWtr 页，之后再更新文件结束时间和文件状态标志。这样的操作顺序保证了文件中数据的完整性，但不足之处是可能造成部分数据（因未达到一页的数据量）的丢失，但由于丢失的这部分数据是整个嵌入式系统掉电前瞬间的部分，这部分数据对目标对象来讲是可以舍弃的。实际验证表明其对后续的数据分析基本没有影响。

在每次写入数据时，软件将判读当前文件的存储情况。若当前文件未达到设定的大小（分区被赋予的最大值），则在数据写入后，文件写位置（页码）ulPageWtr 递增并更新文件结束时间，然后保存文件属性指 FRAM；若文件达到设定的大小，即文件满，则需要在更新文件标识、文件结束时间，并保存文件属性指 FRAM，之后通过“文件号＝（文件号＋1）% MAX _ FLASH _ FILE _ NUM”的算法，计算新的文件号，并从 FRAM 中读出新文件号的文件属性，根据文件标志确定下一个文件的文件号，并进行初始化工作（包括文件标识设置、更新文件创建时间、文件页码置零等），之后保存文件属性至 FRAM，至此形成了一个新的文件，并可以开始保存数据。文件号采用“文件号＝（文件号＋1）% MAX _ FLASH _ FILE _ NUM－1 之间轮回，实现多区循环覆盖使用。其软件设计流程如图 1 所示。

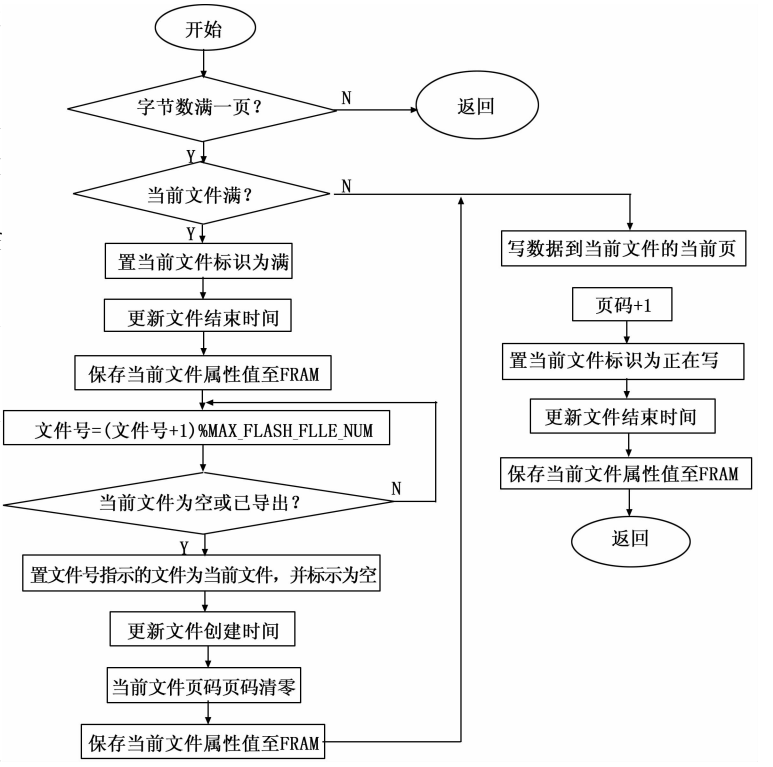

图 1 文件写入流程图

根据上述设计，所需保存的数据将不断地被写入缓存然后转存至 Flash 中，因此文件属性的更新维护每写一页便要进行一次，而且必须在写页数据操作周期间隔内完成，同时需要采用保证数据完整性的存储措施（如每次更新时，FRAM 连续保存三次文件属性，读出时，内容完全相同的两次认为是正确的），这就要求对于文件属性的存储是经常的且不能耗费过多的时间，这也是设计中采取 FRAM 的一个重要原因。

由此，嵌入式系统采集到的数据可自动轮回使用 NAND Flash 存储器的存储空间，达到存储器空间均衡使用的目的，同时，数据写入过程中，文件属性中的数据结束时间、文件写位置以及文件状态标识等不断更新维护，并保存在 FRAM 中，实现了掉电后存储状态的恢复，保证了存储空间使用的连续性。

2 NAND Flash 分区设计思想的应用

该分区设计思想在某型装备传动系统的测试控制、状态监测及故障诊断节点的设计中得到了应用。该节点作为

整车 CAN 总线网络的组成部分，通过 CAN 总线网络进行信息交互，实现传动系统的测试控制、状态监测及诊断，包括故障数据下载和离线分析等。整个节点主要由数据存储单元、温度采集单元、频率采集单元、压力采集单元等组成，要求保存采集控制数据以及故障诊断结果计 60 个通道，存储频率 20Hz，数据保存时间不少于 24 小时。

这里主要讨论数据存储单元，根据要求，仅就数据存储功能来讲，该节点每秒需要保存 20 组数据计 2400 字节（每个通道 2 个字节），其硬件原理如图 2 所示。

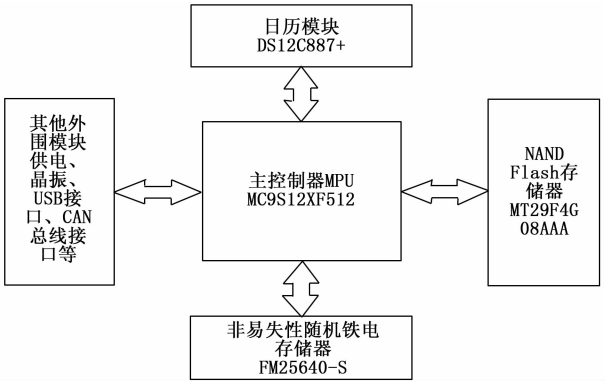

图 2　数据存储单元硬件原理框图

图 2 中，主控制器采用了 MC9S12XF512MLM 单片机，它是 Freescale 公司研发的性价比极高的一款单片机，总线速度可达 50Mhz，价格低、可靠性高；日历芯片采用 DS12C887＋，它可计算到 2100 年前的秒、分、小时、星期、日期、月、秒，其中日历信息并带闰年补偿，自带晶振和锂电池，在没有外部电源的情况下可工作 10 年；非易失性随机铁电存储器采用 FM25640－S，它是一款 64K 位非易失性铁电存储器，可以向 RAM 一样快速读写，支持 SOPI 模式 0&3，结构简单，操作方便；大容量 NAND Flash 存储器选用了 MT29F4G08AAA 存储芯片，该芯片共 4096 块，每块 64 Pages，每页 2，048＋64 bytes。

应用 NAND Flash 分区设计思想，结合大容量 NAND Flash 存储器的指标，对存储器划分为 20 个区，每个分区由 200 个块组成，共计占用存储器 4000 个块空间，剩余空间作为坏块处理机制使用。这样，每个分区由 12800 页组成，存储空间约 26M 字节，由此计算出每个分区可以保存约 3 小时的数据，20 个分区可以保存累计 60 小时的数据，满足了该节点的设计要求，并在实际装车应用中得到了证明。

在嵌入式系统软件设计中，按照 NAND Flash 分区设计思想，定义了文件属性读写函数、NAND Flash 存储器的格式化函数和读写函数、数据存储函数、定时中断服务等。NAND Flash 存储器的格式化操作是建立文件分区的最初操作，分配了文件号及其对应的存储起始地址、文件属性等内容。NAND Flash 存储器的读写函数用于实现对指定页的读写；数据存储函数是对图 1 所示的数据写入流程的实现；定时中断服务用于对通过 CAN 总线接收以及系统自身产生的心跳数据进行定时采集和缓存。

数据的缓存通过一个环形缓冲区的设计实现，在定时中断中将所有要保存的数据进行整理后存入环形缓冲区。数据存储函数根据缓存中的数据量和页码值，计算文件号和页码值，调用 NAND Flash 存储器的写函数将缓存中的数据写入 NAND Flash 存储器的对应页，更新文件属性并通过文件属性写函数保存，完成数据的存储操作。

在上位机软件的配合下，通过 USB 接口实现与嵌入式系统的交互，完成对 NAND Flash 存储器的初始化、数据写入、状态维护、数据下载、数据删除等功能。为方便用户定位数据，上位机软件可以通过文件属性翻译相关信息，以数据起始时间和结束时间来表示具体数据段，方便用户定位具体的关键数据，进行删除、下载等操作，其实际的应用界面如图 3 所示。在实际的应用中，还必须考虑坏块管理以及文件导出和删除等的设计，具体的实现这里不再详述。

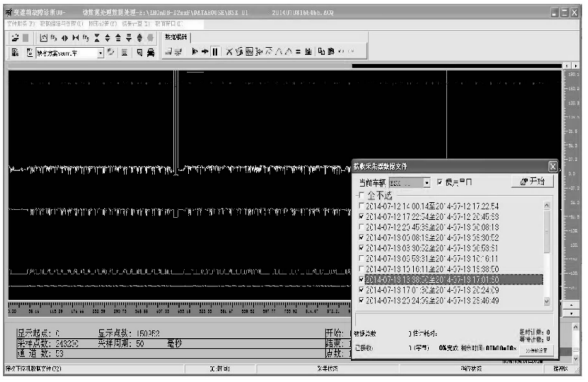

图 3　数据导出与数据编辑界面

3　实验结果与分析

为了验证该分区设计思想，按照某型装备传动系统的测试控制、状态监测及故障诊断节点的设计要求，采用同样的硬件系统（如图 2 所示）设计了车载数据存储记录装置进行实验验证。主要包括以下几个方面：1）数据可选性，主要验证是否可以通过文件属性的元素表征不同的文件，选取指定的文件进行导出、删除等；2）数据使用效率比较；主要目的是看使用该思想后，对于目标数据导出所用的时间，以及后续数据处理时数据检索和处理的效率；3）嵌入式系统执行效率比较，主要目的是验证嵌入式系统对数据存储的完整性、实时性、正确性；4）NAND Flash 存储器使用的均衡性，主要目的是看是否达到 NAND Flash 存储器空间的均衡使用，进而达到延长其寿命的目的；5）软件设计的方便性，主要是验证软件实现的复杂程度。

在同样的硬件系统平台上，在 CodeWarrior5.0 开发环境下设计开发了 FAT 文件管理系统、不分区数据直接存储以及采用本文所述的分区管理思想等三种数据存储管理模式，经多次的实验室验证和实际的装车使用，采用本文所述的分区管理思想设计的车载数据存储记录装置能够通过

（下转第 220 页）