

星载高可靠性嵌入式数据库系统的设计与实现

张东晨,李文新,贾露娟,夏加高

(兰州空间技术物理研究所,甘肃 兰州 730000)

摘要:随着国内航天技术的不断发展,星载平台数据量日渐增多,为了提高星载嵌入式系统对数据的处理能力,在充分考虑现有嵌入式数据库功能的基础上,结合星载嵌入式处理器硬件资源受限的实际情况,提出了一种新型的基于 FatFs 文件系统的星载嵌入式数据库实现方案,大大减少了对系统内存的占用。同时,该方案将应用于通讯领域的汉明码容错技术加以改进并与数据库系统相结合,有效解决了空间环境下星载数据管理系统中极易出现的单粒子事件。经过测试验证表明该方案正确可行,且与传统的嵌入式数据库相比,可以满足星载数据库系统对于高可靠性的要求,弥补了传统数据库系统在数据存储可靠性设计上的不足。

关键词:星载嵌入式系统;数据库;可靠性;文件系统

中图分类号:TP302.1

文献标识码:A

文章编号:1673-629X(2019)12-0093-06

doi:10.3969/j.issn.1673-629X.2019.12.017

Design and Implementation of a Highly Reliable On-board Embedded Database System

ZHANG Dong-chen, LI Wen-xin, JIA Lu-juan, XIA Jia-gao

(Lanzhou Institute of Physics, CAST, Lanzhou 730000, China)

Abstract: With the continuous development of domestic space technology, the amount of data on the satellite platform is increasing. In order to improve the data processing capability of the on-board embedded system, on the basis of fully considering the existing embedded database functionality, combined with the actual situation of limited hardware resources, we put forward a new implementation of the on-board embedded database based on FatFs file system to greatly reduce the system memory usage. At the same time, the program improves Hamming code fault-tolerant technology applied in the field of communication and combines it with the database system to effectively solve the single particle event easily appeared in the space borne data management system. The test proves that the scheme, which is correct and feasible, compared with the traditional embedded database, can meet the high reliability requirements of the on-board database system, and make up for the shortcomings of the traditional database system in the reliability design of data storage.

Key words: satellite-carried embedded system; database; reliability; file system

0 引言

随着国内航天技术的不断发展,在轨任务日趋复杂,航天器的有效载荷数量、种类以及任务形式越来越多,对于卫星上分系统或单机的数据管理水平的要求也不断提高^[1]。为了实现对数据的规范管理,同时缓解数据存取问题对总的星载计算机带来的压力,设计适用于星载嵌入系统的数据库势在必行。对于星载嵌入式系统来讲,由于其特殊的应用背景,与地面嵌入式设备中应用的数据库系统相比,对于可靠性的要求更高^[2-4]。而在现有嵌入式数据库系统中,尚缺乏关

于数据存储可靠性的设计与应用。为应对复杂的空间辐射环境、降低由存储器故障而引起的数据出错概率及其对整星的影响^[5-6],可靠性设计必不可少。

目前,星载存储器的可靠性设计大多采用三模冗余方法对数据进行容错纠错处理^[7-8],但这种方法占用内存资源过大,无法在星载嵌入式系统中高效运行。针对这一问题,文中提出将改进的应用于通讯领域的汉明码容错技术^[9-11]与星载嵌入式数据库系统相结合,以有效地解决星载数据库应对空间单粒子效应时可靠性不足的问题。

收稿日期:2018-12-05

修回日期:2019-04-08

网络出版时间:2019-06-27

基金项目:中国载人航天工程重大专项(RWZY640601);国家自然科学基金(61125101)

作者简介:张东晨(1994-),男,硕士,研究方向为空间电子及星载嵌入式技术应用;李文新,研究员,研究方向为空间电子技术及嵌入式应用。

网络出版地址: <http://kns.cnki.net/kcms/detail/61.1450.TP.20190627.1102.028.html>

1 传统嵌入式数据库体系结构

传统的嵌入式数据库系统如 SQLite^[12]、Empress、Berkeley DB 等,其体系结构大体相同,主要由以下几个子系统^[13]组成:应用 API、数据连接层、SQL 层、内核级 API、数据库引擎、操作系统层和硬件存储介质层。体系结构如图 1 所示^[14]。



图 1 传统嵌入式数据库体系结构

与传统数据库相比,虽然嵌入式数据库已经省去了客户机/服务器运行模式带来的开销,但仅一个操作系统便足以占据整个星载嵌入式平台的内存。以高性能的 BM3803 星载嵌入式处理器^[15]为例,其内存资源仅有 4 兆,即便是最小型的嵌入式操作系统,也无法正常运行。而已有的嵌入式数据库系统无一例外,均需要操作系统的支持,无法满足卫星嵌入式系统对于数据库产品的要求。

2 星载嵌入式数据库体系统的实现

与传统的基于操作系统的嵌入式数据库产品不同,星载嵌入式数据库系统是根据没有独立的服务器进程的 UnQLite 小型非关系型数据库改写而成,其通过将数据直接读取和写入扇区的方式,将具有多个集合的数据库文件存入统一的磁盘文件中。UnQLite 嵌入式数据库的开源代码只提供了 UNIX 和 Windows 两个版本,但由于其需要的外部库和操作系统的支持极小,在经过相关的接口配置和冗余功能裁剪后,仅需要文件系统的支持即可完成对数据的存取管理。具有简单、灵活和高效的特点。

2.1 UnQLite 嵌入式数据库的体系结构

UnQLite 数据库采用的是分层设计的思想,其体系结构如图 2 所示,主要由以下几个模块组成:文件存储层、键/值存储层、通用存储引擎、底层存储引擎、虚拟文件系统和硬件存储介质。

UnQLite 是一个标准的键/值存储数据库,类似于 BerkeleyDB,在存储过程中,每一个键/值对都被视为简单的字节数组,其可以是 ASCII 字符串,二进制数组

甚至是磁盘文件。虚拟文件系统层是设计新的数据库系统时最重要的一环,UnQLite 使用抽象层与操作系统连接,每个操作系统都有自己的实现方式,并通过 unqlite_vfs 函数实现 UnQLite 内核和底层操作系统之间的资源调用。在星载数据库的实现过程中,如何选择合适的文件系统代替原有的操作系统尤为重要。



图 2 UnQLite 数据库体系结构

Unqlite_open 函数中调用了 5 个函数: unqliteCoreInitialize(内核初始化)、SyMemBackend PoolAlloc(后台内存池分配)、SyZero(参数清零)、unqliteInit Database(初始化内存及事务管理)、SyMemBackend Release(后台内存指针释放)、SyMemBackendPoolFree(后台内存池释放)。

其中 unqliteCoreInitialize 又调用了 5 个函数: unqliteExportBuiltinVfs(外部虚拟文件系统指针)、unqlite_lib_config(Unqlite 数据库参数配置)、SySetInit(键值集合结构体初始化)、unqliteExportMemKvStorage(内存键值存储所需函数指针)、unqliteExportDisk KvStorage(磁盘键值存储所需函数指针)。

为了适应星载硬件平台的内存需求,文中在省略操作系统带来的开销过程中,实现了外部虚拟文件系统函数指针所指向的函数,需要编写相应的接口函数,使得 UnQLite 数据库可以在没有操作系统的支持下仅依靠 FatFs 文件系统完成预定功能。完成 UnQLite 与 FatFs 的函数接口的具体方法为实现 FatFs 文件系统 OS 接口对象中的相关定义。其中 OS 接口对象是数据库引擎调用文件系统功能和资源的函数接口,数据库引擎通过 OS 接口对象与 FatFs 文件系统的应用层建立连接。它们包括: fatOpen、fatDelete、fatAccess、fatFullPathname、fatVfs_Mmap、fatVfs_Unmap、fatSleep、fatCurrentTime。其函数功能如表 1 所示。

2.2 星载嵌入式数据库的索引实现

(1) 键/值存储模型。

该数据库使用键/值存储模型完成对数据的索引和存储^[16-17]。与关系型数据库^[18]不同,键/值存储模

型适合存储不涉及过多关系的数据,同时能有效减少读写磁盘的次数,拥有更好的读写性能。数据库的键/值存储功能函数接口如表 2 所示。

表 1 OS 接口对象

函数名称	函数功能
fatOpen()	根据数据库宏定义配置执行打开文件的操作
fatDelete()	根据数据库宏定义配置执行删除文件的操作 (如果定义了文件夹同步删除的宏开关,则删除对应的文件夹)
fatAccess()	根据数据库宏定义查询文件属性(是否存在、是否只读、是否可读写)
fatFullPathname()	将相对路径转为绝对路径
fatTmpDir()	可裁剪
fatVfs_Mmap	可剪裁
fatVfs_Unmap	可剪裁
fatSleep()	进行延时操作
fatCurrentTime()	获取当前系统时间
fatGetLastError()	可裁剪

表 2 键/值存储函数接口

接口名称	函数功能
unqlite_kv_store	将记录存储在数据库中(将新记录写入数据库,如果记录不存在,则创建该记录,否则替换原有记录)
unqlite_kv_append	将数据附加到数据库记录(将新记录写入数据库,如果记录不存在,则创建该记录,否则将新的数据块加到旧块的末尾)
unqlite_kv_fetch	从数据库中获取记录,并将其内容复制到用户提供的缓冲区
unqlite_kv_delete	从数据库中删除特定记录
unqlite_kv_config	配置存储引擎属性

(2) 游标迭代器。

游标提供了一种机制,可以通过该机制迭代数据库中的记录。使用游标同样可以搜索,获取,移动和删除数据库记录。在使用游标之前,必须首先使用 unqlite_kv_cursor_init() 分配新的游标句柄。这通常是应用程序发出的第一个 UnQLite 游标 API 调用,并且是使用游标的先决条件。完成后,必须调用 unqlite_kv_cursor_release() 以通过游标释放已经分配的资源,从而避免内存泄漏。如果需要迭代数据库中的记录,从第一个记录到最后一个记录,只需调用 unqlite_kv_cursor_first_entry() 并连续调用 unqlite_kv_cursor_next_entry(),直到它返回 UNQLITE_OK 以外的值。与此同时,可以调用 unqlite_kv_cursor_valid_entry() 来检查光标是否指向有效记录(这将在有效时返回 1,否则返

回 0)。还可以使用游标搜索记录并从那里开始迭代过程。

2.3 虚拟文件系统的接口配置

在文中所述的数据库实现过程中,选用 FatFs 文件系统替代已有的操作系统层。FatFs 文件系统^[19]的系统结构如图 3 所示。在该方案设计中,FatFs^[20]的应用层即为改写后的 UnQLite 嵌入式数据库。FatFs 模块与物理设备完全分离,其底层存储设备驱动控制模块需要在实现过程中根据具体的硬件平台进行设计。其平台独立、易于在各硬件平台间进行移植,且指令编译和程序代码占用内存空间很小,非常适合应用于硬件资源受限的星载嵌入式平台上。

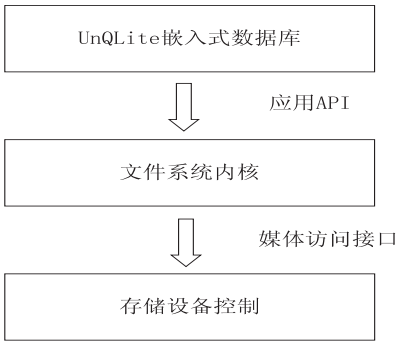


图 3 FatFs 文件系统结构示意图

FatFs 总共有 5 个文件,分别是 ff.c、ff.h、diskio.c、diskio.h、integer.h。通常情况下,ff.h 和 interger.h 在移植到星载嵌入式平台的过程中不需要进行改动。移植工作主要更改的是 diskio.c,配置工作主要更改 ff.h 和 diskio.h。

在星载数据库系统的运行过程中,FatFs 文件系统提供的功能主要包括:打开/创建文件、关闭一个打开的文件、从文件中读取数据、将数据写入文件、刷新缓存的数据等。其具体的函数功能如表 3 所示。

3 可靠性设计

与传统的汉明码在进行数据校验时应满足一定的位数要求不同,假设星载嵌入式数据库系统的汉明码原始数据信息位长为 m 位,校验位长度为 r 位,则汉明码在编码解码过程中只要满足以下条件即可。

$$2^K - 1 \geq m + r \tag{1}$$

(1) 数据编码原理。

首先校验位必须设置在 1,2,4,8,16 等这些 2 的整数次幂位。由于 UnQLite 数据库的每个单元均由两个字节的地址组成,即每 16 位地址存储一条原始记录,但由于需要加入校验位,因此在写入数据时每 8 位地址存储一条原始记录,其余空间用以存储校验位信息。

其次,根据偶(奇)性测试的原理来确定校验位的

内容。在确定校验位的内容前,先要对数据位进行分组。分组原理为:对所有位的序号位,即对表中 1,2,3,4,5,6,7,8,9,10,11,12……位,进行二进制转换并设置为逆序。第一组数据是经过二进制转换后,数据位右边起第 1 位总是 1 的数,第二组数据是经过二进制转换后,数据位右边起第 2 位总是 1 的数,第三组数

据是经过二进制转换后,数据位右边起第 3 位总是 1 的数,第四组数据是经过二进制转换后,数据位右边起第 4 位总是 1 的数。数据分组后,分别按照各组的顺序对各校验位进行填写,各位校验位顺序对应各组数据的偶(奇)性测试,若数组中“1”的个数为奇数,则填“1”,若数组中“1”的个数为偶数,则填写“0”。

表 3 文件系统函数接口

	函数名称	函数功能
应用接口 函数功能	f_Open ()	打开/创建文件
	f_Close ()	关闭一个打开的文件
	f_Read ()	从文件中读取数据
	f_Write ()	将数据写入文件
	f_Sleek ()	刷新缓存数据,即打开文件对象中的文件读/写指针,指向要在下一个读/写操作过程中需要读/写的数据字节
	f_Sync ()	刷新缓存的数据
媒体访问接口 函数功能	f_SectorSize	返回默认文件分区大小
	disk_status	获取设备状态
	disk_initialize	初始化设备
	disk_read	读取扇区
	disk_write	写扇区
	disk_ioctl	控制设备指定特性和除读/写外的其他功能
	get_fattime	获取当前时间

其功能由汉明码写入函数 hmfat_Write 实现,在写入数据时按汉明码原理进行编码处理,创建两份文件。一份写入的是原始数据,一份写入的是编码后的数据。

数据写入分组如表 4 所示,其中第 1、2、4 和第 8 位为校验位。

表 4 数据写入分组

分组	数据位(首行为数据位,次行为二进制转换后的数据位)					
第一组	1	3	5	7	9	11
	0001	0011	0101	0111	1001	1011
第二组	2	3	6	7	10	11
	0010	0011	0110	0111	1010	1011
第三组	4	5	6	7	12	空
	0100	0101	0110	0111	1100	
第四组	8	9	10	11	12	空
	1000	1001	1010	1011	1100	

(2)数据解码原理。
初始化数据库系统后,调用 hmfat_Read 函数对数据库存储的数据进行解码输出,每次读取两个字节数据,即 16 位数据(其中包括 8 位原始数据位,4 位校验位)。

首先对读取的两个字节数据进行偶(奇)性测试,检验写入的数据是否正确,如果正确,则剔除校验位,获取最终 8 位有效数据。偶性测试原理为:由于写入数据时进行了相应的编码处理,在校验位填写了适当

的“0”或者“1”,确保每次校验过程中偶性测试都满足得到偶数个“1”。所以,在读取数据时,对各组数据再次进行偶性测试,如果发现某次这组数据未能得到偶数个“1”,则说明有对应数据位发生错误。对比四组分组数据的出错情况,即可找到出错数据的具体位置,进行“0”、“1”翻转即可。

4 实验验证

文中采用星载嵌入式系统中常用的 BM3803 嵌入

式处理器搭载 NAND FLASH 存储器为硬件实验平台,但由于 BM3803 处理器暂时没有配套的显示器,所以文中采用以 BM3803 为平台的微振动数据采集系统作为星载嵌入式数据库的测试用例。通过对航天器地面测试平台的微振动数据的高频采集、存储和读取,来验证星载高可靠嵌入式数据库实现方案的可行性和正确性。其系统框图如图 4 所示。

该系统以 625 Hz 的采样高速频率进行微振动信号采集,然后将 3 路振动信号写入 UnQLite 数据库,以

键-值格式文件存入 FLASH 芯片,检索数据库中的 3 路传感器数据(X,Y,Z)后,通过 RS232 串口通信将数据上传至 PC 上位机。其数据采集结果如表 5 所示。其中加速度值 G 由分层值经式 2 和式 3 计算得出。分层值即为记录在数据库中的原始数据。 X 为分层值,常数 32 767 为分层值满量程, V 为电压值。

$$V = V_{\text{ref}} * \frac{X}{32\ 767}$$

(2)

$$G = V * \gamma$$

(3)

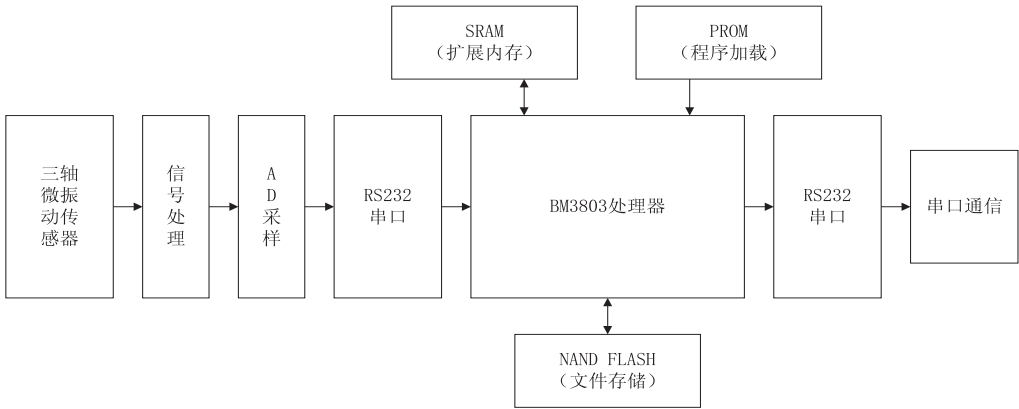


图 4 微振动系统采集系统框图

表 5 数据采集结果

通道名称	次数	分层值 (-32 766 ~ 32 767)	G / mg	人为造错 (原始数据位 1 位出错)	实际数据
X	1	-8 232	2.010	否	2.010
X	2	-8 100	1.980	否	1.980
.....
Y	1	-8 725	-2.130	否	-2.130
Y	2	-8 580	-2.095	是	-2.095
.....
Z	1	5 150	1.257	否	1.257
Z	2	5 652	1.380	否	1.380
.....

其中 Z 轴传感器约 20 秒微振动数据采集结果如图 5 所示。

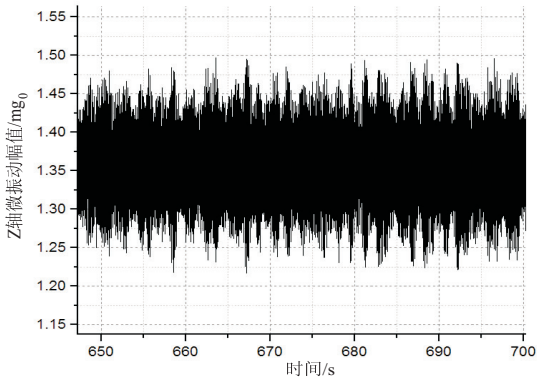


图 5 微振动数据测试结果

从 Z 轴微振动数据的检索测试结果来看,数据库进行高频存取后的数据与实际振动测量数据结果相同,在单个数据位出错的情况下,能自动检错并纠正错误信息,说明该微振动测量系统完成了对星载数据库系统的检索查询功能,实现了嵌入式数据库读和写操作,并且具备较高的数据存储可靠性。与其他实时多任务系统(uC/OS 、 linux 、 FreeRTOS)集成的嵌入式数据系统相比,基于 UnQLite 的星载嵌入式数据库与文件系统直接对接,系统结构更简单,对外部资源利用更为合理,可以满足卫星数据的存取要求。

5 结束语

虽然目前世界上主流的数据库产品很多,但还没

有将其应用在航天器嵌入式系统上的案例,其主要原因在于星载嵌入式平台大多采用结构简单但可靠性高的处理器,此类处理器内存资源受限,无法直接运行已有的数据库产品。同时,在现有的数据库产品中,均缺乏对于提高数据存储可靠性的设计,这使得数据出错无法自动恢复,不适合在复杂的空间环境中应用。文中提出的高可靠性星载嵌入式数据库系统很好地解决了上述两个问题。该数据库接口配置规范简单,使其不受平台限制,极易在不同硬件间移植。经过反复的实验验证,该方案正确可行,同时为航天器数据管理这一领域的相关研究具有一定的参考价值。

参考文献:

- [1] STUBENRAUCH C J,ROSSOW W B,KINNE S,et al. Assessment of global cloud datasets from satellites:project and database initiated by the GEWEX radiation panel[J]. Bulletin of the American Meteorological Society,2013,94(7):1031-1049.
- [2] 宋琪,邹业楠,李 珊,等. 卫星固态存储器数据容错设计与机制[J]. 国防科技大学学报,2016,38(1):101-106.
- [3] 贾露娟,李文新,夏加高,等. 星载嵌入式容错文件系统的设计与实现[J]. 计算机技术与发展,2015,25(10):49-53.
- [4] 申 奥. 高可靠并行星载计算机软件容错技术研究[D]. 上海:上海交通大学,2013.
- [5] LEGUIZAMO C P,MORI K. Autonomous coordination in the information allocation in distributed database systems for assurance [C]//IEEE workshop on distributed computing systems. [s. l.]:IEEE,2003.
- [6] PENGRA B, LONG J, DAHAL D, et al. A global reference database from very high resolution commercial satellite data and methodology for application to Landsat derived 30 m continuous field tree cover data[J]. Remote Sensing of Environment,2015,165:234-248.
- [7] 陈 州,倪 明. 三模冗余系统的可靠性与安全性分析[J]. 计算机工程,2012,38(14):239-241.
- [8] 张 超,赵 伟,刘 峥. 基于 FPGA 的三模冗余容错技术研究[J]. 现代电子技术,2011,34(5):167-171.
- [9] 辛 英. 汉明码纠错检错能力分析与应用[J]. 盐城工学院学报:自然科学版,2008,21(1):34-36.
- [10] 沈云付,潘 磊. 三值汉明码检错纠错原理和方法[J]. 计算机学报,2015,38(8):1648-1655.
- [11] 甘家宝. 汉明码校验原理解析[J]. 微型电脑应用,2007,23(1):58-60.
- [12] CHEN Dai, HAN Xudong, WANG Wei. Use of SQLite on embedded system [C]//International conference on intelligent computing & cognitive informatics. Kuala Lumpur, Malaysia: IEEE,2010:210-213.
- [13] SEIPEL D, BOEHM A M, FRÖHLICH M. JSquash: source code analysis of embedded database applications for determining Sql statements [C]//International conference on applications of declarative programming and knowledge management. Évora, Portugal: Springer,2011:153-169.
- [14] 陆慧娟. 嵌入式数据库原理与应用[M]. 北京:清华大学出版社,2013.
- [15] 詹盼盼,郭廷源,高建军,等. 基于 BM3803 处理器的即插即用星载计算机系统设计[J]. 航天器工程,2013,22(6):92-96.
- [16] 赵 恒. 面向键值数据库应用的混合存储系统设计与实现 [D]. 武汉:华中科技大学,2012.
- [17] 黄贤立. NoSQL 非关系型数据库的发展及应用初探 [J]. 福建电脑,2010,26(7):30.
- [18] 潘农菲. GIS 的空间数据在关系型数据库的实现理论及应用技术[J]. 计算机应用研究,2002,19(2):92-94.
- [19] 张 涛,左谨平,马华玲. FatFs 在 32 位微控制器 STM32 上的移植[J]. 电子技术,2010,47(3):25-27.
- [20] 秦 伟. STM32 的 FatFS 在数据采集系统中的应用[J]. 单片机与嵌入式系统应用,2015(6):55-58.