计算机测量与控制.2009.17(12)
Computer Measurement & Control

文章编号:1671-4598(2009)12-2516-02 中图分类号:TP334 文献标识码:B

基于 VxWorks 操作系统下大硬盘的实现

侯小鹏,李家志,吴岳松
(中国舰船研究设计中心,湖北 武汉 430064)

摘要:在实时嵌入式操作系统中,VxWorks 因其高可靠性高实时性而得到广泛应用,基于 VxWorks5.5 版本开始支持高版本 MS-DOS 文件系统,从而给我们挂接大硬盘提供了前提;以实际工作中挂接大硬盘过程,从原理上分析了 VxWorks 下硬盘的管理机制,并总结出 VxWorks 操作系统中挂接大硬盘所采取的相关配置情况及具体实现,从而得出在 VxWorks5.5 下通过正确的配置,可以实现大硬盘的挂接,同时也给出挂接后硬盘测试程序。

关键词:实时操作系统;Vxworks;块设备;CBIO 接口

Running Big Capacity HD in VxWorks OS

Hou Xiaopeng, Li Jiazhi, Wu Yuesong
(China Ship Development and Design Center, Wuhan 430064, China)

Abstract: In real-time embedded operation system, VxWorks is applied in many fields for its high reliability and the real-time characteristic. For the edition of VxWorks 5.5 had supported high edition of MS-DOS file system, which give us a way to solve the problem of running big capacity HD in it. Based on the actual works, the paper analysed the management of HD in VxWorks and introduced how to configurate and run the OS with big capacity HD, So get an conclusion that we can run big capacity HD in VxWorks OS by proper configuration. At the same time, corresponding program testing HD is supplied.

Key words: real-time operation system; Vxworks; block device; CBIO

0 引言

VxWorks 操作系统中,提供了 dosFs 文件系统以兼容 MS-DOS 文件,支持的 DOS 版本最高为 DOS6.22。然而在 DOS6.22 系统下,仅支持 FAT16 格式的文件系统,这种格式的文件系统限定了硬盘最大只能为 2G,这对于需要存储大量数据的应用程序来说已经成为存储瓶颈。随着嵌入式系统中数据量的日益增大,这种瓶颈效应益发突出,应对的措施只能是将大硬盘分为多个小于等于 2G 的分区,这显然不是理想的解决之道。随着 VxWorks5.5 版本的推出,其特色之一就是开始支持高版本的 MS-DOS 系统,而高版本 DOS 系统就开始支持 FAT32 格式的文件系统,这就为解决大硬盘问题提供了前提。

1 VxWorks 简介

VxWorks 是一个运行在目标机上的高性能、可裁减、功能强大而且比较复杂的嵌入式实时操作系统,包括了进程管理、存储管理、设备管理、文件系统管理、网络协议及系统应用等几个部分。VxWorks 只占用了很小的存储空间,并可高度裁减,保证了系统能以较高的效率运行。它以其良好的可靠性和卓越的实时性被广泛地应用在通信、军事、航空、航天等高精尖技术及实时性要求极高的领域中,如卫星通讯、军事演习、弹道制导、飞机导航等。在美国的 F-16、FA-18 战斗机、B-2 隐形轰炸机和爱国者导弹上,1997 年 4 月在火星表面登陆的火星探路者上也使用了 VxWorks。[5]

VxWorks 操作系统是美国 WindRiver System 公司(以下简称风河公司,即 WRS 公司)于 1983 年设计开发的一种嵌入式实时操作系统(RTOS),是 TornadoII 嵌入式开发环境的关键组成部分。良好的持续发展能力、高性能的内核以及友好的用户开发环境,在嵌入式实时操作系统领域逐渐占据一席之地。首先,它十分灵活,具有多达 1800 个功能强大的应用程序接口(API);其次,它适用面广,可以适用于从最简单到最复杂的产品设计;再次,它可靠性高,可以用于从防抱死刹车系统到星际探索的关键任务;最后,适用性强,可以用于所有的流行的 CPU 平台。VxWorks 嵌入式实时操作系统包括微内核 wind、高级的网络支持、强有力的文件系统和 I/O 管理、C++和其他标准支持等核心功能。这些核心功能还可以与 WindRiver 公司的其他产品以及 320 个 WindRiver 公司的合作伙伴的产品联合使用。

VxWorks 实时操作系统由 400 多个相对独立的、短小精炼的目标模块组成,用户可根据需要选择适当模块来裁剪和配置系统,这有效地保证了系统的安全性和可靠性。系统的链接器可按应用的需要自动链接一些目标模块。这样,通过目标模块之间的按需组合,可得到许多满足功能需求的应用。[4]

2 VxWorks 下硬盘的管理

在 VxWorks 下,硬盘不能直接与 I/O 系统打交道,在其与 IO 系统之间必须有文件系统,如 dosFs、rt11Fs、rawFs 或 tapeFs 等,这种层次关系允许同一个块设备上使用不同的文件系统,为了实现上层软件的设备无关性,所有的块设备驱动都遵循统一的接口规范,该接口不以单独的软件层存在,而是体现在一个统一的数据结构上(BLK_DEV)。该结构由具体的设备实例化,用来代表该设备。对于上层软件来说,与块设备的接口就是这个统一结构,通过具体的结构实例来操作对应的设备。该结构中包含了统一的函数接口,函数具体的实现由各块设备驱动程序完成。为了方便块设备的创建,各块设备都遵

收稿日期:2009-06-09; 修回日期:2009-07-31。

作者简介:侯小鹏,男,硕士,工程师,主要从事软件设计工作的研究。

循类似的接口和类似的初始化过程。如 xxxDrv 用于设备相关初始化，应先于设备实际操作前调用，一般由系统帮助完成。xxxDrv 只执行一次，影响整个设备控制器，而其他后继的操作只影响特殊的设备。xxxDrv 完成的功能包括初始化硬件、分配和初始化设备数据结构、创建互斥信号量和初始化中断等。xxxDevCreate 用来创建设备实例，当创建块设备实例时，设备没有名称与其相联，只有建立文件系统后才有名称，块设备再经过 CBIO 封装由文件系统使用。CBIO（Cached Block I/O）即块设备缓存接口，在 VxWorks5.5 下，块设备要实现与文件系统的挂接，必须转换为 CBIO 接口，这是文件系统的向下接口。cbioLib 块设备缓存 IO 库（Cached Block IO）用于提供缓存块输入输出编程接口，同时也为其他的 CBIO 模块提供基础支持，还提供一个基本的 CBIO 模块，为块设备提供接口封装，封装设备可以直接挂接文件系统，且只有极小的内存消耗。“CBIO - CBIO 设备”上下接口都遵循 CBIO API。

dcacheCbio 通过 CBIO 接口实现存储盘 Cache 机制，将经常使用的块存储在内存中以提高效率，主要针对 DOS 文件系统使用。Cache 不知道具体文件格式，只按块为对象操作。dcacheCbio 可用于大多数块设备，下层可以挂接 CBIO 接口或块设备接口，当直接和块设备连接时，会用 cbioLib 提供的基本 CBIO wrapper 封装块设备 API。Cache 按块为单位组成，按最近使用（MRU）顺序组织为链表。当需要空间保存新块时，删除链表尾部的 LRU 块，除了这些通常的 Cache 块外，Cache 分出部分空间用于“大缓存”操作（Burst），一般为 Cache 大小的 1/4，最大可以为 64K。

dpartCbio 库利用 CBIO 接口实现多分区的管理，支持各个分区创建独立文件系统。该分区管理库需要一个外部库支持，如 usrFdiskPartLib，用于解析特殊的存储盘分区表。当和 dcacheCbio 库堆叠使用时，建议将其放在 dcacheCbio 上面，作为主 CBIO 设备。调用 dpartDevCreate 应先调用 dcacheDevCreate。这样 dcahceCbio 层就可以为整个存储盘服务，而不是只对单分区服务。支持用 MSDOS 的 FDISK 创建的分区表格式，这需要 usrFdiskPartLib 库的支持，该库在“\target\src\usr\usrFdiskPartLib.c”中实现，可以用于计算机的多分区硬盘的挂接，最多在一个磁盘上支持（C～Z）24 各分区。

3　VxWorks 下硬盘的配置

当创建了一个 bootable 工程后，则系统为我们生成 PrjConfig.c 文件，在此文件里面，使用 void usrIosExtraInit (void) 函数调用 usrAtaInit () 函数（位于“\target\src\config\usrAta.c”中，这个函数根据 config.h 中所配置的 ATA_RESOURCE < ataResouces [] > 的参数，执行硬盘初始化工作），接下来会执行语句

```
if (strcmp(DOSFS_NAMES_ATA_PRIMARY_MASTER, ""))
usrAtaConfig (0, 0, DOSFS_NAMES_ATA_PRIMARY_MASTER);
if (strcmp(DOSFS_NAMES_ATA_PRIMARY_SLAVE, "" ))
  usrAtaConfig (0, 1, DOSFS_NAMES_ATA_PRIMARY_SLAVE);
if (strcmp(DOSFS_NAMES_ATA_SECONDARY_MASTER, "" ))
usrAtaConfig (1, 0, DOSFS_NAMES_ATA_SECONDARY_MASTER);
if (strcmp(DOSFS_NAMES_ATA_SECONDARY_SLAVE, "" ))
usrAtaConfig (1, 1, DOSFS_NAMES_ATA_SECONDARY_SLAVE);
```

因此，需根据硬盘的具体挂接情况（可在目标机 BIOS 里面查看），修改宏定义，如：硬盘挂接为 SECONDARY_MASTER，且分为两个区，则修改宏定义为 define DOSFS_NAMES_ATA_SECONDARY_MASTER“/ata0C, ata0D”。在 usrAtaConfig 函数里面，先使用 ataDevCreate () 函数创在整个磁盘上建块设备，接下来通过 cbio = dcacheDevCreate ((CBIO_DEV_ID) pBlkDev, NULL, ATA_CACHE_SIZE, freePtr)；语句来创建一个 CBIO 层的磁盘缓存，接下来通过 masterCbio = dpartDevCreate (cbio, numPart, usrFdiskPartRead)；创建分区管理器，接着通过 stat = dosFsDevCreate (devName [pn], dpartPartGet (masterCbio, pn), NUM_DOSFS_FILES, NONE)；创建 dosFs 文件系统，从而实现硬盘与文件系统的挂接。值得注意的是 bootrom 启动时并不执行该函数，所以启动时默认的是 ata0 = 0，0，因此要报 bootrom.sys 和 vxworks 同时放在 C 盘，一旦 vxworks 启动则会识别出 D 盘。还有根据实际情况，在 config.h 文件里面将 ATA Logical Type 由 ATA ——PCMCIA 修改为 IDE_LOCAl，当完成这些修改后，在工程中 shell 下可以使用 devs 命令查看，正常下会显示出所挂接的两个硬盘。

4　大硬盘挂接实例

硬件组成：舰载加固机，里面配有 Pentium3 的 CPU 板一块，多功能板一块，4G 容量的电子盘挂在调试板上，挂接过程如下：

(1) 在 Windows 系统下，使用 Powermangic 分为两个区：基本 Dos 分区 C：100M，扩展 Dos 分区 D：3900M。C 盘格式化为 Fat16，D 盘格式化为 Fat32，将此硬盘挂接在 SECONDARY_MASTER.

(2) 接着在 Config.h 文件里面配置启动行：采用网络启动方式，则配置为：" fei (0, 0) host: vxWorks h = 192.10.100.1 e = 192.10.100.5 u = target pw = target".

(3) 配置硬盘 define INCLUDE_ATA，修改硬盘参数 define ATA 1_CTRL_TYPE (ATA_PCMCIA) 改为 define ATA 1_CTRL_TYPE (IDE_LOCAL)，修改 ATA_TYPE < ataTypes [] [] > 里面的参数，define ATA_CTRL1_DRV0_CYL (761)，define ATA_CTRL1_DRV0_HDS (8)，define ATA_CTRL1_DRV0_SPT (39).

(4) 配置网络 define INCLUDE_END，define INCLUDE_FEI_END.

(5) 修改非易失性内存大小：

```
if (SYS_WARM_TYPE == SYS_WARM_BIOS)
 define NV_RAM_SIZE (NONE)
else
 define NV_RAM_SIZE (NONE)
endif
```

此时，根据这些配置生成 bootrom.sys 可实现 Vxworks 系统的引导，并能正常加载 Vxworks 映象。可以使用以下代

据检测结果调整工作状态。由于开关频率较高，对 A/D 采样有噪声干扰，所以采用递推加权平均值滤波的软件滤波方法、软件冗余和软件陷阱等方法来提高系统的抗干扰能力[5-6]。

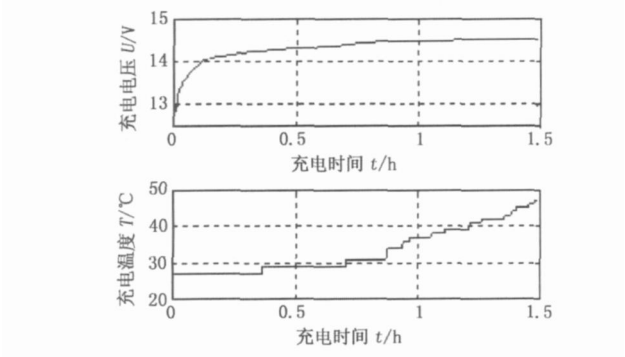

图 8 电池端电压和温度曲线图

5 实验结果

通过对 12V/1800mA 的镍氢电池进行多次充电实验，根据记录的实验数据绘制出电池端电压、电池温度以及充电电流的变化曲线，如图 8 所示。由充电电流曲线可以看出：(1) 利用模糊控制的充电方法充电时间基本在 1.5 h 左右；(2) 在整个充电过程中，充电电流较好地跟随了马斯以最低析气率为前提给出的可接受的最佳充电电流，使充电电流不进入析气区，节省了电能，有效防止了电池过热；(3) 70 min 时电池端电压 U 达到最大值，20 min 内可检测到 $0\Delta V$ 现象，检测到 $0\Delta V$ 现象后充电器用 50 mA 的电流对镍氢电池进行小电流浮充，从而可以有效避免镍氢电池过充。

6 结论

以可接受的最佳充电电流曲线为基础的模糊控制智能充电系统与采用传统充电方法的充电系统比较，具有充电快速、效率高、电池温升低等特点，保障了充电电池的安全，从而延长电池寿命，与传统充电器相比有实时监测电池状态和自适应功能。该数字化智能充电系统具有实际应用价值。

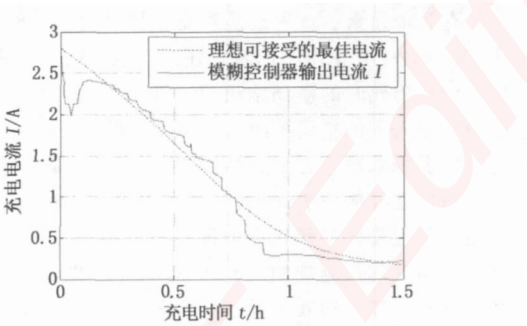

图 9 充电电流变化曲线图

参考文献：
[1] 王鸿麟，景占荣. 通信基础电源 [M]. 西安：西安电子科技大学出版社，2002.
[2] 高 田，景占荣，羊 彦. 蓄电池组快速充电模糊控制技术的研究 [J]. 计算机仿真，2006，(10)：236-238.
[3] 徐大诚，等. 微型计算机控制技术及应用 [M]. 北京：高等教育出版社，2003.
[4] 谢季坚，刘承平. 模糊数学方法及其应用 [M]. 武汉：华中科技大学出版社，2005.
[5] 刘 洋，等. 基于 ATMEGA48V 单片机的电动车驱动系统设计 [J]. 计算机测量与控制，2008，16 (12)：1860-1862.
[6] 万光毅，等. 单片机实验与实践教程 [M]. 北京：北京航空航天大学出版社，2003.

(上接第 2517 页)

码进行硬盘的读写测试。

```
    include <VxWorks.h>
    include <stdio.h>
    include <iolib.h>
    include <stat.h>
void main (void)
{
    int fdc;
    int fdd;
    int saveFile;
    char buf[20] = "Just For Test!";
    char diskFree[12];
    struct statfs pstatfs;
    fdc = open ("/ata0C/net.dat", O_CREATE | O_RDWR,
0644);
    fdc = open ("/ata0D/net.dat", O_CREATE | O_RDWR,
0644);
    fstatfs (fdc, &pstatfs);
    printf ("Totle block in file system: %ld\n", pstatfs.f_blocks);
    printf ("Free block in file system: %ld\n", pstatfs.f_bfree);
}
```

程序输入将会在屏幕上打印出当前硬盘的实际已用字节数及可用字节数。

5 总结

VxWorks5.5 相比 5.4 版本，在性能上进行了许多优化，同时对于 I/O 设备的管理更加完善，使我们对程序的开发变得更为简单。通过实际挂接大硬盘的过程及目前的运行状况来看，Vxworks 对于大硬盘的支持是很完善的。

参考文献：
[1] 房少军. 嵌入式实时操作系统 VxWorks 及其开发环境 Tornado [M]. 北京：电子工业出版社，2008.
[2] VxWorks Programmer's Guide 5.5 [Z]. Wind River Systems, Inc.
[3] Tornado User's Guide 2.2 [Z]. Wind River Systems, Inc.
[4] 陈智育，温彦军，陈 琪. VxWorks 程序开发实践 [M]. 北京：北京人民邮电出版社，2004.
[5] 梁 永，孟 桥. 嵌入式操作系统 VxWorks 中的多媒体组件 WindML2.0 [J]. 现代计算机，2002，10 (149).