

基于 PC/104 的实时嵌入式 Linux 系统开发

冯 敏, 樊晓光, 褚文奎
(空军工程大学 工程学院, 陕西 西安 713800)

摘要:结合 PC/104 模块 EmCORE - i316VL/C400 的主要特征,构建了一个实时嵌入式 Linux 系统,并就构建过程中内核定制、引导程序安装、GNU 工具链开发和获取实时性支持等方面做了详细阐述。简要分析了文件系统的选择原则。

关键词: PC/104;实时嵌入;linux

中图分类号: TP316.8 **文献标识码:** A **文章编号:** 1009 - 3516(2006)01 - 0084 - 03

1 系统开发平台与规划

图 1 是构建的系统开发平台,配备 256 MB 的 DRAM 作为系统内存和内装 Redhat linux 9.0 的 4.3 GB 的 IDE 硬盘。此外,还采用 128 MB 的 flash 盘存储临时文件,128 MB 的 CF 卡存储开发的目标系统,以及监视器用来调试开发的中间状态。图 1 中虚线右侧是开发期间的辅助设备。

嵌入式系统主机/目标板开发规划一般分为 3 种:连接式、可抽换式和独立式^[1]。一般而言连接式规划适用于网络开发及后期调试,可抽换式规划适用于开发初期却疲于存储装置的实体转移,而独立式规划较适用于以 PC 为主的高阶嵌入式系统(如高可用性系统)。在分析比较了三种规划的基础上,我们综合了可抽换式和独立式规划的优点,提出新的规划方案,见图 2。在这个规划中,主机与目标板共用同一 CPU,通过借用 IDE 硬盘空间将建构开发得到的实时嵌入式系统移植到 CF 卡中。图 2 中虚线箭头表示移植方向。

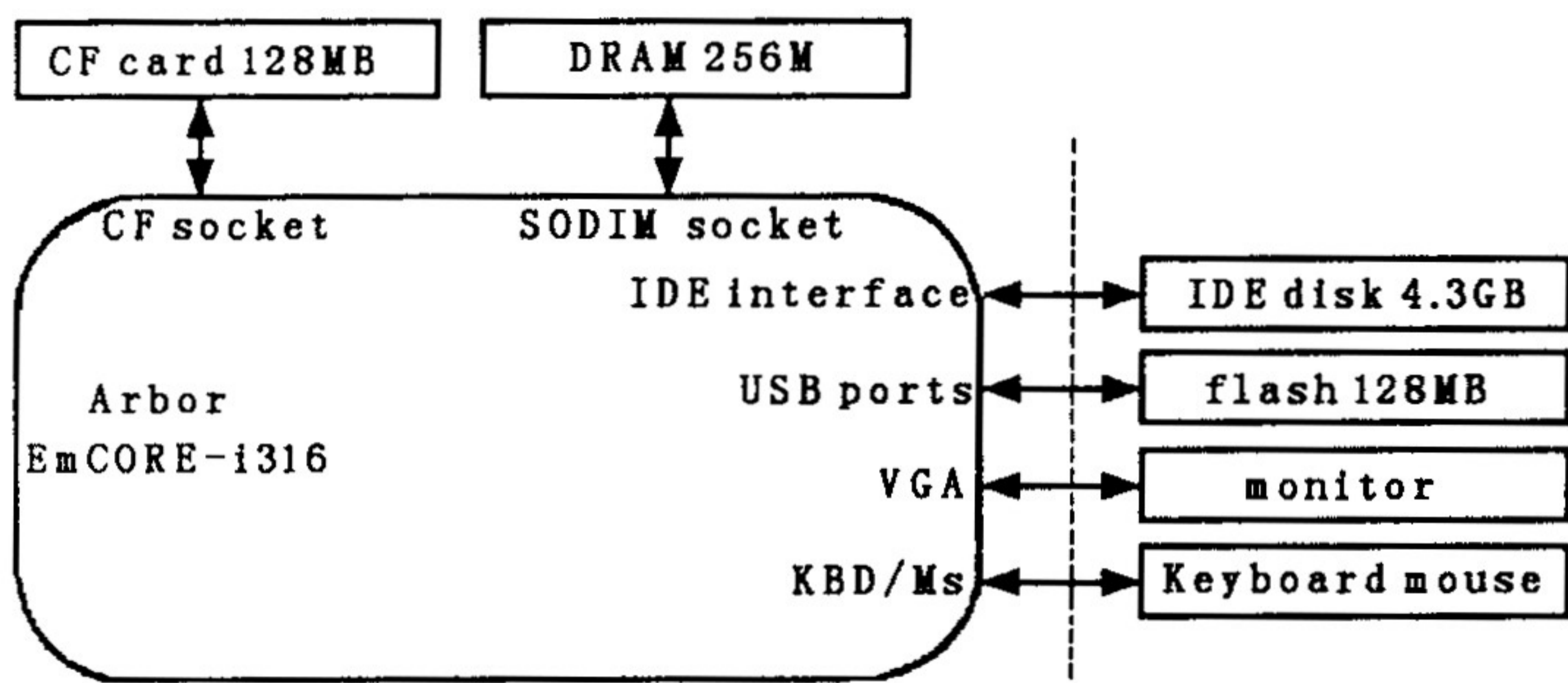


图 1 系统开发平台

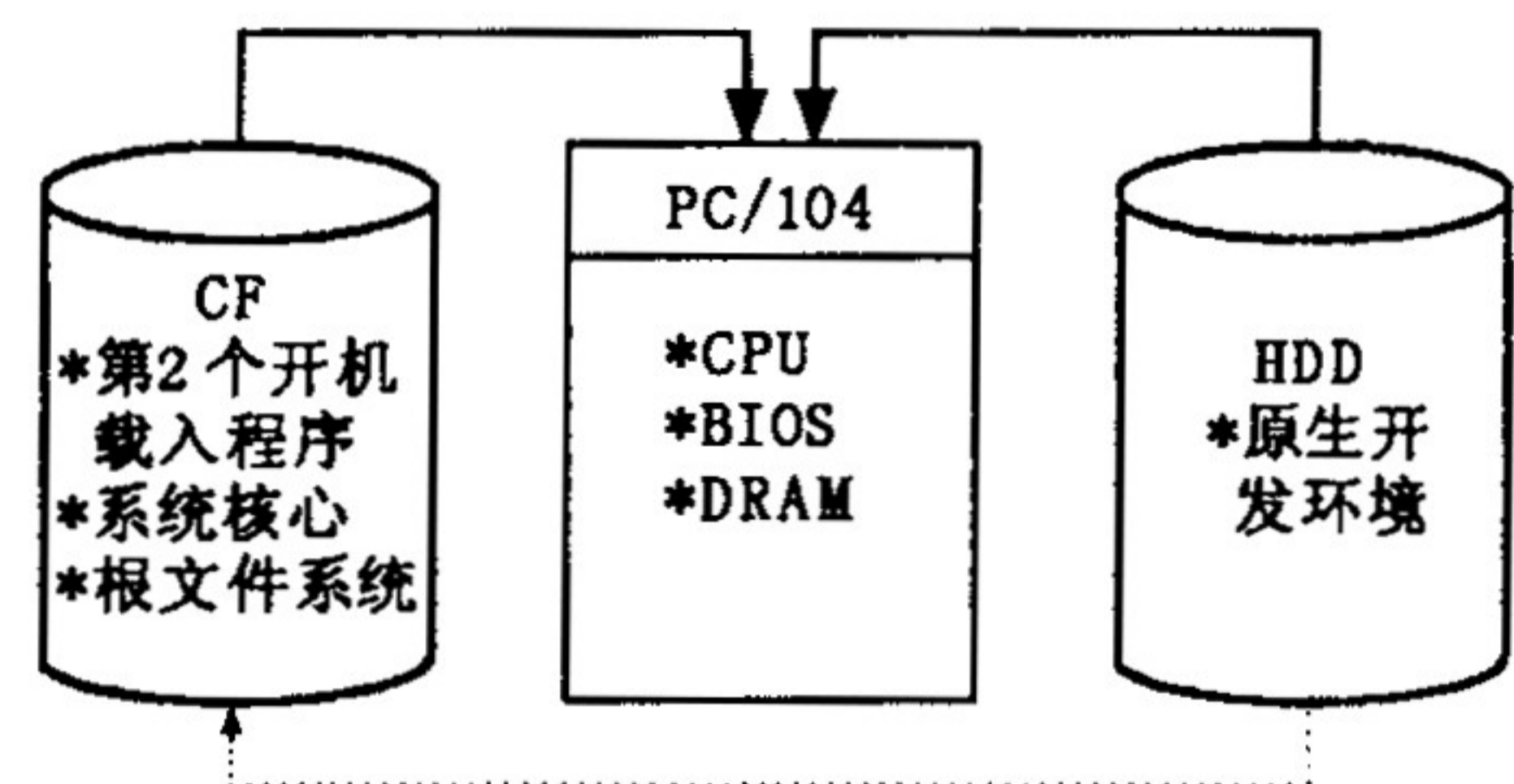


图 2 系统开发规划

2 实时嵌入式系统的构建

2.1 定制 linux 内核

鉴于目标板中只有 128 M 的 CF 外部存储设备,需要对 linux 内核加以裁减定制,以使内核更为精简、高效及稳定。

收稿日期:2005 - 05 - 27

基金项目:军队科研基金资助项目

作者简介:冯 敏(1978 -),河南开封人,硕士生,主要从事分布式系统与智能检测。

为了提高目标系统的实时性能,在定制之前,还需要根据实际应用补丁内核程序,例如采用 kernel_patch - 2.4.24 - rtl3.2 - pre3 补丁以使 linux - 2.4.24 内核支持实时性能^[2-3]。根据目标应用,我们选择 x86 架构、网络支持、ext2 文件系统等完成 linux 内核定制。此后将编译安装得到的 vmlinuz - rtl3.2 - 2.4.24 - pre3、initrd - rtl3.2 - 2.4.24 - pre3.img、System - rtl3.2 - 2.4.24 - pre3 存储到 $\$ \{PRJROOT\}/boot$ 目录下,用以 grub 引导。

2.2 安装引导程序

引导程序一般要完成初始化系统硬件、实现各种通信协议、启动系统内核等工作。安装系统引导程序有助于嵌入式 linux 系统在出现意外当机或死机时可以快速自动重新进入系统,增强系统的健壮性。

常见的引导程序有 LILO、GRUB、U-BOOT 等。一般而言,U-BOOT 用于 PowerPC、ARM 架构居多,而如果使用 LILO,在每次改变配置或者重新安装内核时就需重新运行它以使新的配置或内核生效。相对于 LILO,GRUB 就具有更好的灵活性。GRUB 可以将任意一内核安装到所需的任意一硬盘分区上,并且在编译安装新内核时系统会自动修改 GRUB 菜单。我们的目标系统采用了 GRUB 引导程序。

在安装 GRUB 之前,还需考虑 PC/104 CPU 模块如何读取 CF 卡。如果 CPU 模块通过 USB CF 读卡机存取时,CF 设备在主机上会被当作 SCSI 磁盘设备,其对应的安装结点是 /dev/sd*,而当通过 CF-to-IDE 或 CF-to-PCMCIA 适配器读取时,其在目标板上会被当作 IDE 磁盘设备,对应安装结点是 /dev/hd*。Em-CORE-i316VL/C400 PC/104 CPU 模块提供一个 CF-to-IDE 适配器插槽。所以如果将 CF 用作第二个磁盘设备,那么安装结点应是 /dev/hd1。安装 GRUB 如下:

```
/* 应用原生程序 grub */
#/sbin/grub
grub > root (hd1,0)
grub > setup (hd1)
```

安装后,需要编写目标板所需的 $\$ \{PRJROOT\}/boot/grub/grub.conf$ 文件,此时需注意根文件系统在第一个磁盘上。因为在目标系统中只有 CF 卡设备,没有其他磁盘。

2.3 开发编译环境

由于目标板与主机合一,所以这里不存在开发嵌入式系统常见的交叉编译问题。但是为了保证建构的目标系统可自行创建和编译应用程序,仍需要 GNU 开发工具链,主要包括 binutils 包中的 as、ar、ld、nm、objcopy、strip、objdump,gcc 包中的 gcc、g++、am,以及 glibc 或者 uClibc 库等。相对而言,glibc 或者 uClibc 库都很大,目标板应用系统无需如此多的库函数,需要进行裁减。此处裁减的一个艺术在于:先在硬盘上编译好 GNU 开发工具链,再将 binutils 和 gcc 可执行文件存储到 CF 卡上,然后使用 ldd 命令查看这些可执行文件需要的支持库,再将这些支持库拷贝到 $\$ \{PRJROOT\}/lib$ 、 $\$ \{PRJROOT\}/usr/lib$ 中。例如:

```
#ldd $ {PRJROOT}/usr/bin/gcc
libc.so.6 => /lib/tls/libc.so.6 (0x42000000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
#strip -unneeded -strip -o \
$ {PRJROOT}/lib/tls/glibc.so.6 /lib/tls/glibc.so.6
#strip -unneeded -strip -o \
$ {PRJROOT}/lib/ld-linux.so.2 /lib/ld-linux.so.2
```

此过程虽然比较费时但可以为嵌入式系统节约许多宝贵空间。

2.4 获取实时性支持

当前实现系统的实时性策略主要有:直接修改 linux 调度器,并在内核中创建可抢占点;将 linux 作为一个线程运行在一个小的可抢占的实时系统中^[4]。直接修改 linux 内核难度较大,而且调试过程比较繁琐。开发一个实时核心对 linux 内核修改较少,又能充分利用其现有的功能,是获取实时性能支持的一条捷径。

在遵守 FSMLabs 实验室协议下下载实时模块 rlinux - 3.2 - pre3^[2]。将它移植在 linux - 2.4.24 核心底层,对所有中断进行初始化处理。如果中断由实时任务产生,就交由实时核心处理,反之则交由 linux 内核处理。在这个过程中,实时核心实际上是把 linux 内核当作它的一个低优先级的实时任务进行处理。由于 linux 核心优先级最低,且可被其他实时任务抢占,所以运行在用户态下的 linux 应用程序不会影响运行在内

核态下的实时任务的实时性能。其系统构架见图 3。

类似于 linux 内核定制的方法,rtlinux - 3.2 - pre3 实时核心也可以使用 make menuconfig 命令进行直观的裁减。一般而言,rtlinux - 3.2 - pre3 目录下绝大多数的文件选项如 include、drivers、schedulers、system、fifos、main 等都是必要的,使用默认方式即可。

2.5 文件系统的选择

linux 支持的文件系统主要有 CRAMFS、JFFS、JFFS2 以及 NFTL 上的 Ext2、Ext3, RAM disk 上的 Ext2 等。为选择最佳的文件系统或其组合,一般从能否写入、是否具有永久性、是否具备断电可靠度、是否支持压缩以及是否存放于 RAM 中来考察^[1],见表 1。

表 1 文件系统的特性

文件系统	永久	断电可靠	可写	经过压缩	RAM 中
CRAMFS			否	是	否
JFFS	是	是	是	否	否
JFFS2	是	是	是	是	否
NFTL 上的 ext2	是	否	是	否	否
NFTL 上的 ext3	是	是	是	否	否
RAMdisk 上的 ext2	否	否	是	否	是

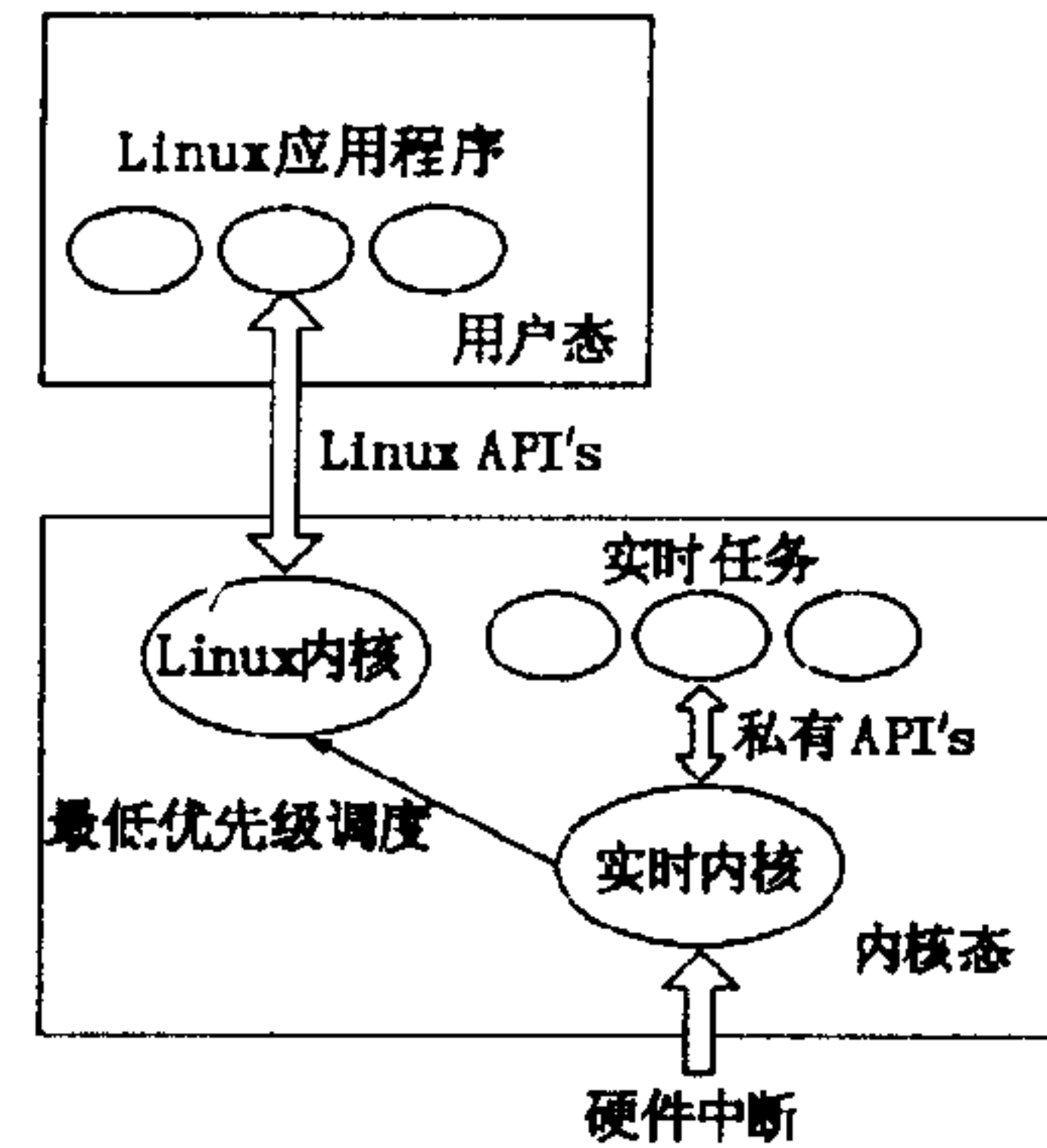


图3 基于双内核的实时系统构架

在目标系统中,还需要开发应用程序,并将其写入到 CF 卡中,这就要求文件系统必须是可写的。128 MB 的 CF 卡对于命令行状态的系统而言足够应用,综合考虑,我们选择 NFTL 上的 ext2 作为文件系统。

3 结束语

构建基于 PC/104 CPU 模块的实时嵌入式 linux 系统,是将开源软件引入到测控领域而迈出的重要一步。经过系统测试,该系统的最大中断延时为 23 μs,调度精度达到 46 μs,很好地解决了实时性问题,具有超低功耗、集成度高、便携性好和实时性佳等特点。

但是,该系统也同时存在着可操作性差、直观性不强等缺点。这是由于为了实现系统的微型化我们没有引入图形用户界面而仅仅开发了基于命令行的应用程序。下一步的研究需要移植 GUI 并在此基础之上开发图形用户程序。然而一旦这样做,就需要重新审视系统的实时性能,在可操作性和实时性之间寻找一个均衡,这是问题的重点,也是难点。

参考文献:

[1] Karim Yaghmour. Building Embedded LINUX Systems[M]. O'Reilly & Associates, Inc. 2003

[2] FSMLabs. rtlinux - 3.2 - pre3. tar. bz2[EB/OL]. <http://www.rtlinux.org/2004-11-4>.

[3] Linuz. etc. linux - 2.4.24. tar. bz2[EB/OL]. <http://www.kernel.org/pub/,2004-11-4>.

[4] Rick Lehrbaum. The real - time linux software quick reference guide [EB/OL]. <http://www.linuxdevices.com/articles/AT8073314981.html/,2004-11-4>.

(编辑:姚树峰)

(下转第 94 页)