浙江大学
ZHEJIANG UNIVERSITY

嵌入式系统

第9课 嵌入式文件系统设计

王总辉
zhwang@zju.edu.cn

http://course.zju.edu.cn

提纲

- 嵌入式文件系统简介
- Linux文件系统框架
- 本地文件系统
- 网络文件系统
- 根文件系统设计

提纲

- 嵌入式文件系统简介
- Linux文件系统框架
- 本地文件系统
- 网络文件系统
- 根文件系统设计

嵌入式文件系统简介

- 普通文件系统
  - 管理文件，提供文件系统API
  - 管理各种设备，支持对设备和文件的操作
- 嵌入式文件系统
  - 为嵌入式系统的设计目的服务
  - 设计目标各不相同

嵌入式文件系统设计目标（1）

- 使用简单方便
- 安全可靠
- 实时响应
- 接口标准的开放性和可移植性
- 可伸缩性和可配置性

嵌入式文件系统设计目标（2）

- 开放的体系结构
- 资源有效性
- 功能完整性
- 支持热插拔
- 支持多种文件类型

常见的嵌入式文件系统

- 基于Flash的文件系统
  - 基于Block Device的文件系统：FAT、EXT2
  - 基于MTD的文件系统：JFFS2、YAFFS2、UBIFS
- 内存文件系统
  - Ramdisk
  - Ramfs/tmpfs
- 网络文件系统
  - Samba
  - NFS

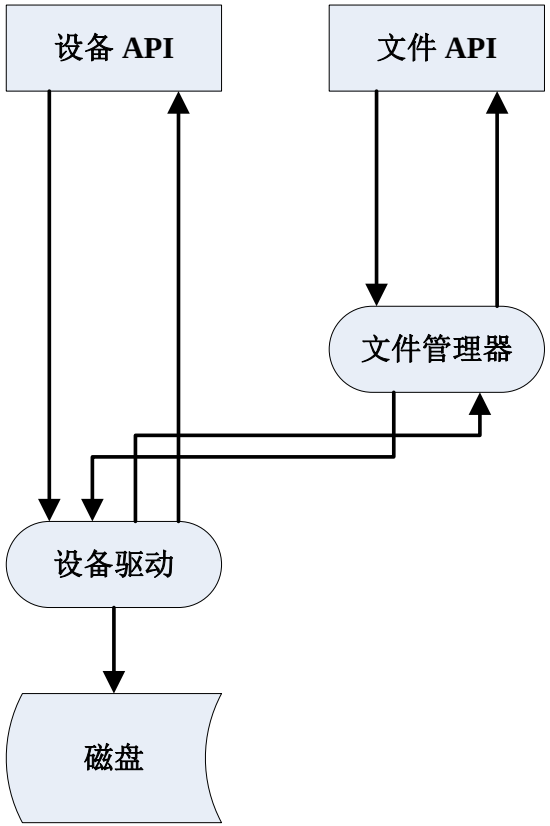
提纲

- 嵌入式文件系统简介
- Linux文件系统框架
- 本地文件系统
- 网络文件系统
- 根文件系统设计

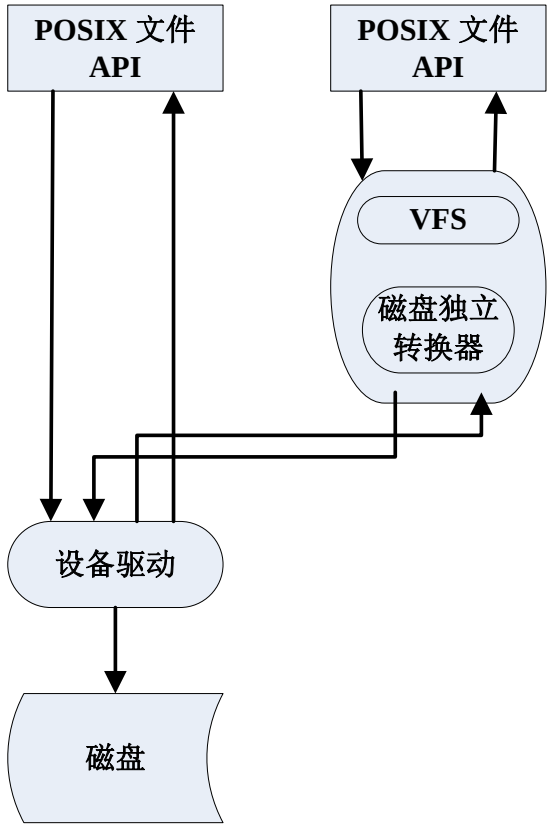
Linux文件系统框架

- Linux文件系统具有良好的扩展性
  - 支持ext、ext2、ext3、ext4、xia、vfat、minix、msdos、umsdos、 proc、smb、ncp、iso9660、sysv、hpfs、affs、ufs
- 现代操作系统都提供多种访问存储设备的方法，Linux文件系统有两条独立控制设备驱动
  - 通过设备驱动的接口
  - 通过文件管理器接口

Linux文件系统框架特点

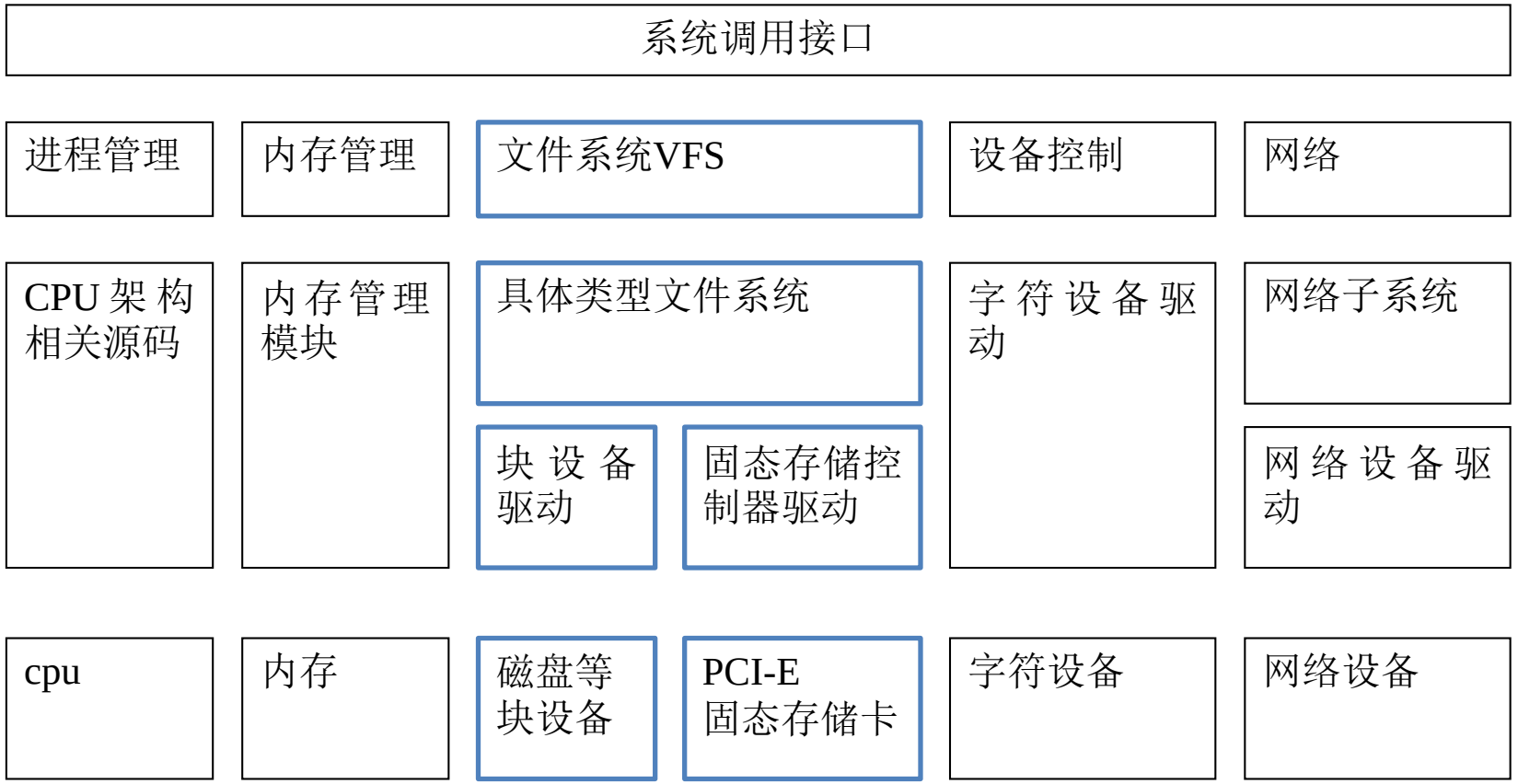


（a）传统文件系统

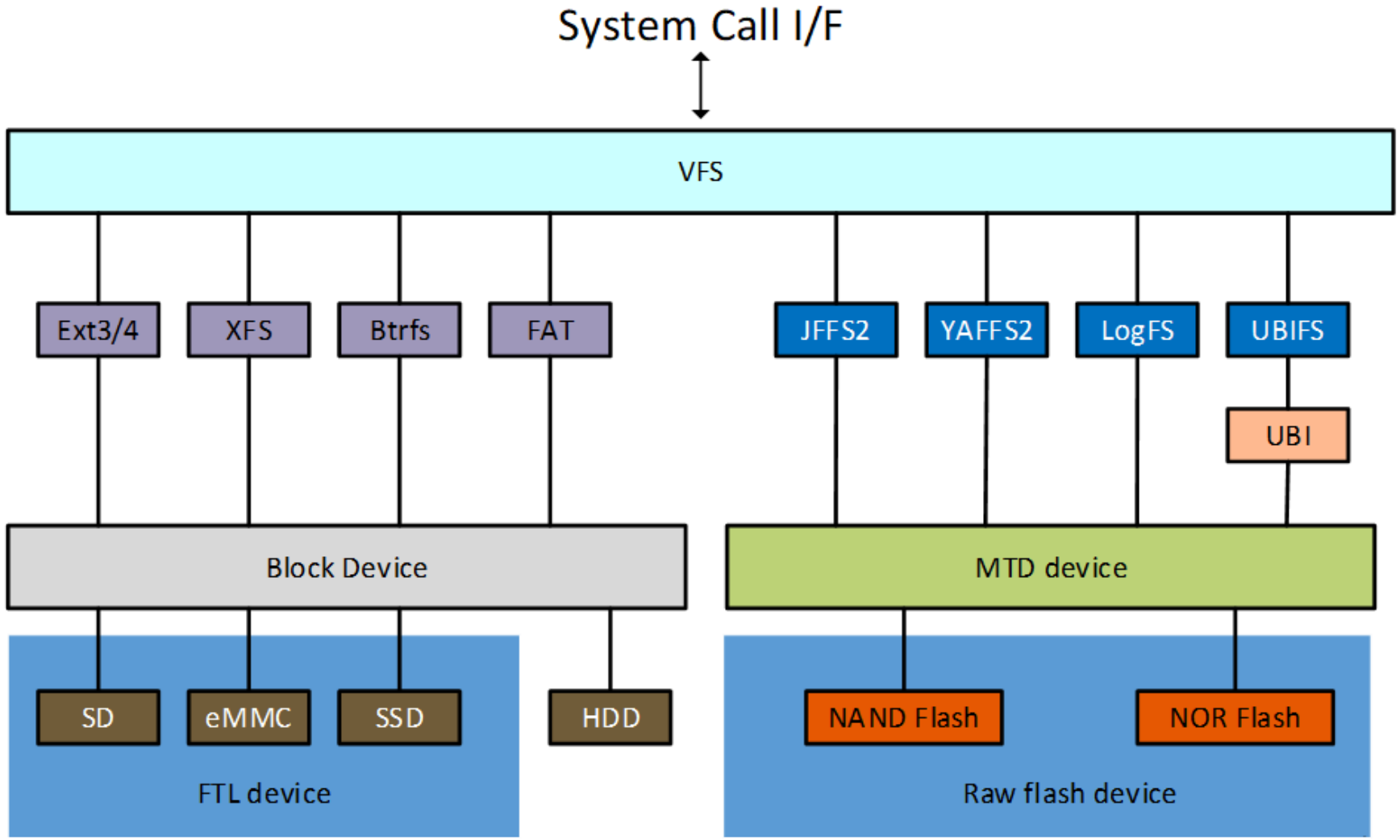


（b）Linux文件系统

文件系统在Linux中位置



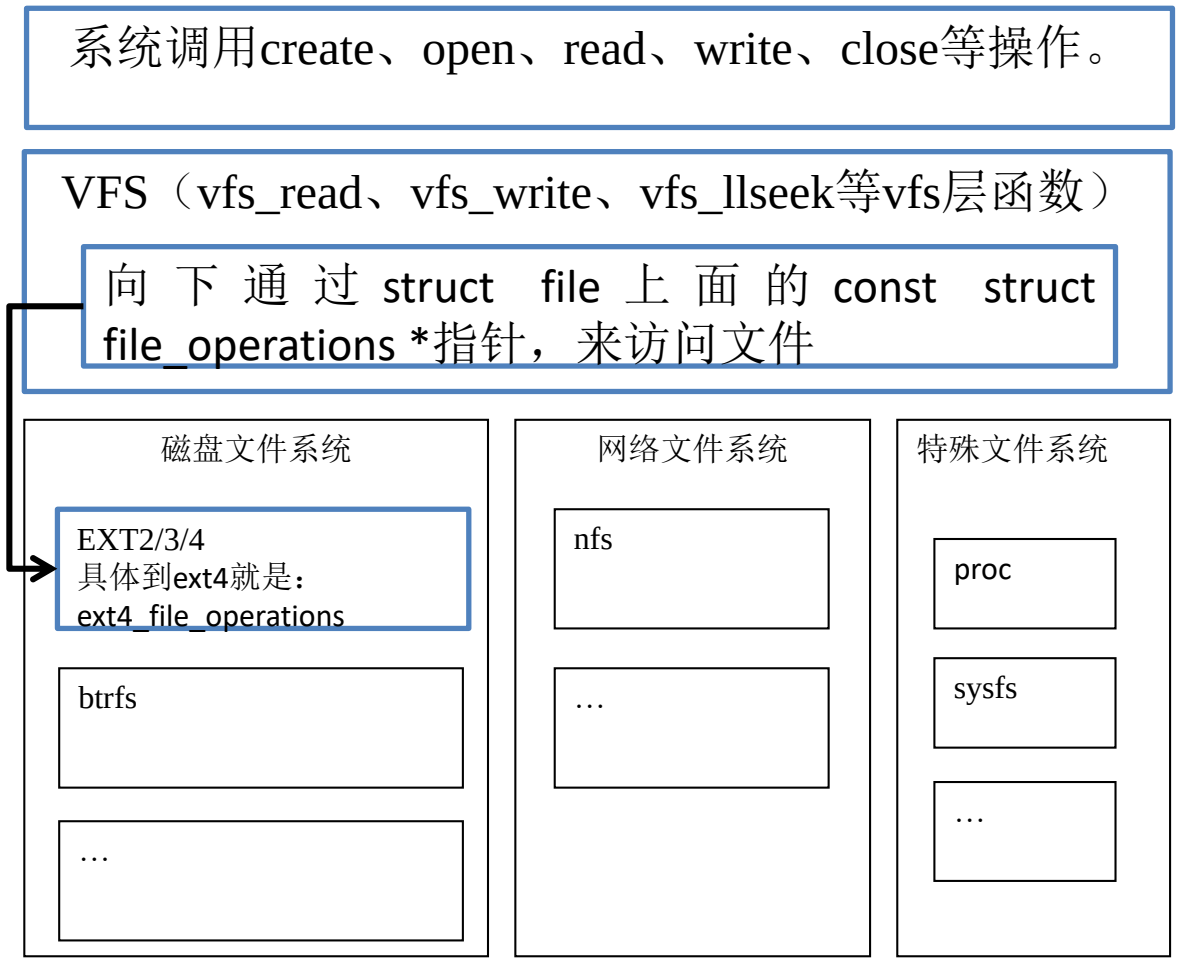
嵌入式文件系统


System Call I/F
VFS
Ext3/4
XFS
Btrfs
FAT
JFFS2
YAFFS2
LogFS
UBIFS
UBI
Block Device
MTD device
SD
eMMC
SSD
HDD
NAND Flash
NOR Flash
FTL device
Raw flash device

文件系统模块层次图

文件系统模块层次图

每个文件系统实现（比如ext2/ext3/ext4、btrfs 等等）导出一组通用接口，供 VFS 使用，和上层VFS对接。



提纲

- 嵌入式文件系统简介
- Linux文件系统框架
- **本地文件系统**
- 网络文件系统
- 根文件系统设计

本地文件系统

- 基于Block Device的文件系统
  - ext2文件系统
- 基于MTD的文件系统
  - JFFS2文件系统
  - YAFFS2文件系统
  - UBIFS文件系统
- 基于内存的文件系统
  - Ramfs/tmpfs

Flash存储器

- Flash存储器（是否带有Flash控制器）
  - Flash芯片设备（Raw Flash device，也叫MTD设备），比如NOR Flash、NAND Flash等
  - 带Flash控制器的设备（Flash Translation Layer device，FTL设备），比如SD、eMMC、SSD、USB大容量存储设备等

	MTD (NAND)	FTL
Interface	Direct interface to NAND controller (SoC)	Block device interface (USB mass storage, SD, ...)
NAND flash compatibility	Dependent on NAND controller (SoC)	Dependent on FTL controller
Bad blocks, Bit-flips, Endurance	Not managed →Upper layer must manage them.	Managed by FTL controller →Upper layer does not have to manage them.
File systems	JFFS2, YAFFS2, UBIFS, ...	FAT, EXT2, EXT3, ...

ext2文件系统 — 优点

- ext2fs支持达4TB容量（ext是2G）
- ext2fs文件名称最长可以到1012个字符
- 当创建文件系统时，可以选择逻辑块大小（1024/2048/4096字节）
- ext2fs实现快速符号链接，无需分配数据块，并提升性能

ext2文件系统 — 缺点

- ext2fs是针对IDE设备设计，逻辑块是512B的倍数，不太适合扇区大小的闪存设备
- ext2fs没有提供对基于扇区的擦除/写操作的良好管理，比如擦除一个字节需复制整扇区到RAM中
- ext2fs在出现电源故障时不防崩溃
- ext2fs不支持损耗平衡，缩短闪存寿命

JFFS文件系统（1）

- JFFS文件系统最早是由瑞典Axis Communications公司基于Linux2.0的内核为嵌入式系统开发的文件系统
- JFFS2是RedHat公司基于JFFS开发的闪存文件系统，可用在Linux,uCLinux中。
- JFFS2文件系统
  - JFFS2文件系统主要针对NOR FLASH设计，是一种基于Flash的日志文件系统。
  - JFFS2的底层驱动主要完成文件系统对Flash芯片的访问控制，如读、写、擦除操作
  - 不适合用于NAND闪存,因为NAND闪存的容量一般较大,这样导致jffs为维护日志节点所占用的内存空间迅速增大;jffs文件系统在挂载时需要扫描整个FLASH的内容,以找出所有的日志节点,建立文件结构,对于大容量的NAND闪存会耗费大量时间

JFFS文件系统（2）

- 优点
  - 可读写、支持数据压缩
  - 基于哈希表的日志型文件系统
  - 提供了崩溃/掉电安全保护
  - 提供“写平衡”支持
- 缺点
  - 当文件系统已满或接近满时，因为垃圾收集的关系，使jffs2的运行速度大大放慢

YAFFS文件系统（1）

- YAFFS针对NAND FLASH设计，和JFFS相比它减少了一些功能(例如不支持数据压缩)，所以速度更快，挂载时间很短，对内存的占用较小
- 跨平台的文件系统，除了Linux和eCos，还支持WinCE, pSOS和ThreadX等
- 采用多策略垃圾回收算法，能够提高垃圾回收的效率和公平性，达到损耗平衡的目的
- 自带NAND芯片驱动，并为嵌入式系统提供了直接访问文件系统的API

YASSF文件系统（2）

- YAFFS2是YAFFS的改进版本
  - YAFFS仅支持小页(512B) NAND闪存， YAFFS2则可支持大页(2KB) NAND闪存
  - YAFFS2在内存空间占用、垃圾回收速度、读/写速度等方面均有大幅提升

UBIFS文件系统（1）

- UBIFS（Unsorted Block Image File System）最早在2006年由IBM与Nokia的工程师Thomas Gleixner，Artem Bityutskiy所设计，专门为了解决MTD设备所遇到的瓶颈，是JFFS2文件系统的下一代
- 由于NAND Flash容量的暴涨，YAFFS等皆无法再去控制NAND Flash的空间。UBIFS通过子系统UBI处理与MTD device之间的动作
- 与JFFS2一样，UBIFS 建构于MTD device 之上，因而与一般的block device不兼容

UBIFS文件系统（2）

- UBIFS在设计与性能上均较YAFFS2、JFFS2更适合NAND Flash
  - UBIFS 支持 write-back，其写入的数据会被cache，直到有必要写入时才写到Flash，大地降低分散小区块数量并提高I/O效率。
  - UBIFS文件系统目录存储在Flash上，UBIFS mount时不需要scan整个Flash的数据来重新创建文件目录。
  - 支持on-the-flight压缩文件数据，而且可选择性压缩部份文件。
  - UBIFS使用日志（journal），可减少对Flash index的更新频率。

Flash文件系统选择

- 当前主流的选择是UBIFS和YAFFS2，如果是非Linux系统可以选用移植性较好的YAFFS2

	System requirement	JFFS2	YAFFS2	LogFS	UBIFS
1	Boot time	Poor	Good	Excellent	Good
2	I/O performance	Good	Good	Fair	Excellent
3	Resource usage	Fair	Excellent	Good	Fair
4	NAND device life expectancy	Good	Fair	N/A	Excellent
5	Tolerance for unexpected power-off	Good	Good	Poor	Good
6	Integrated in mainline	Yes	No	No	Yes

Romfs

- Romfs文件系统是一种简单的、紧凑的、只读的文件系统
- 不支持动态擦写保存，按顺序存放数据
- 支持应用程序以XIP(eXecute In Place，片内运行)方式运行，在系统运行时，节省RAM空间
- uClinux系统通常采用Romfs文件系统

Cramfs：Compressed ROM File System

- Cramfs是Linux的创始人 LinusTorvalds参与开发的一种只读的压缩文件系统，基于MTD驱动程序
- 在cramfs文件系统中，每一页(4KB)被单独压缩，可以随机页访问，其压缩比高达2:1,为嵌入式系统节省大量的Flash存储空间，使系统可通过更低容量的FLASH存储相同的文件，从而降低系统成本
- Cramfs文件系统以压缩方式存储，在运行时解压缩，所以不支持应用程序以XIP方式运行
- 只读属性同时是它的一大缺陷，使得用户无法对其内容对进扩充

Ramdisk

- Ramdisk是将一部分固定大小的内存当作分区来使用，并非一个实际的文件系统
- 是一种将实际的文件系统装入内存的机制，可以作为根文件系统。将一些经常被访问而又不会更改的文件（如只读的根文件系统）通过Ramdisk放在内存中，可以明显地提高系统的性能
- 在Linux的启动阶段，initrd提供了一套机制，可以将内核映像和根文件系统一起载入内存

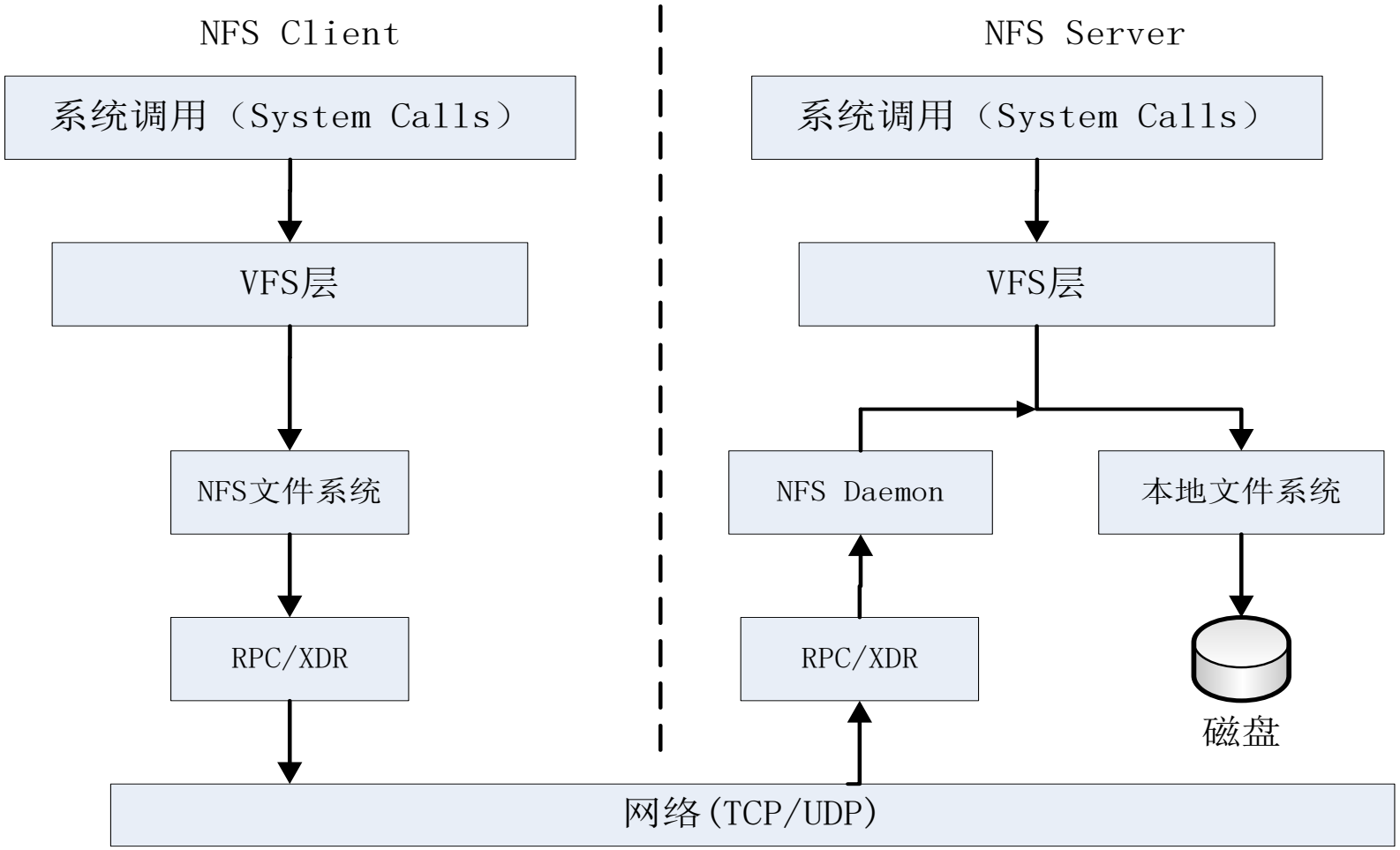
Ramfs文件系统

- Ramfs是Linus Torvalds开发的一种基于内存的文件系统，工作于虚拟文件系统(VFS)层，可以创建多个，在创建时可以指定其最大能使用的内存大小
- 读/写操作发生在RAM中，可用来存储一些临时性或经常要修改的数据，例如/tmp和/var目录，这样既避免了对Flash存储器的读写损耗，也提高了数据读写速度
- 不能格式化，文件系统大小可随所含文件内容大小变化
- 当系统重新引导时会丢失所有数据

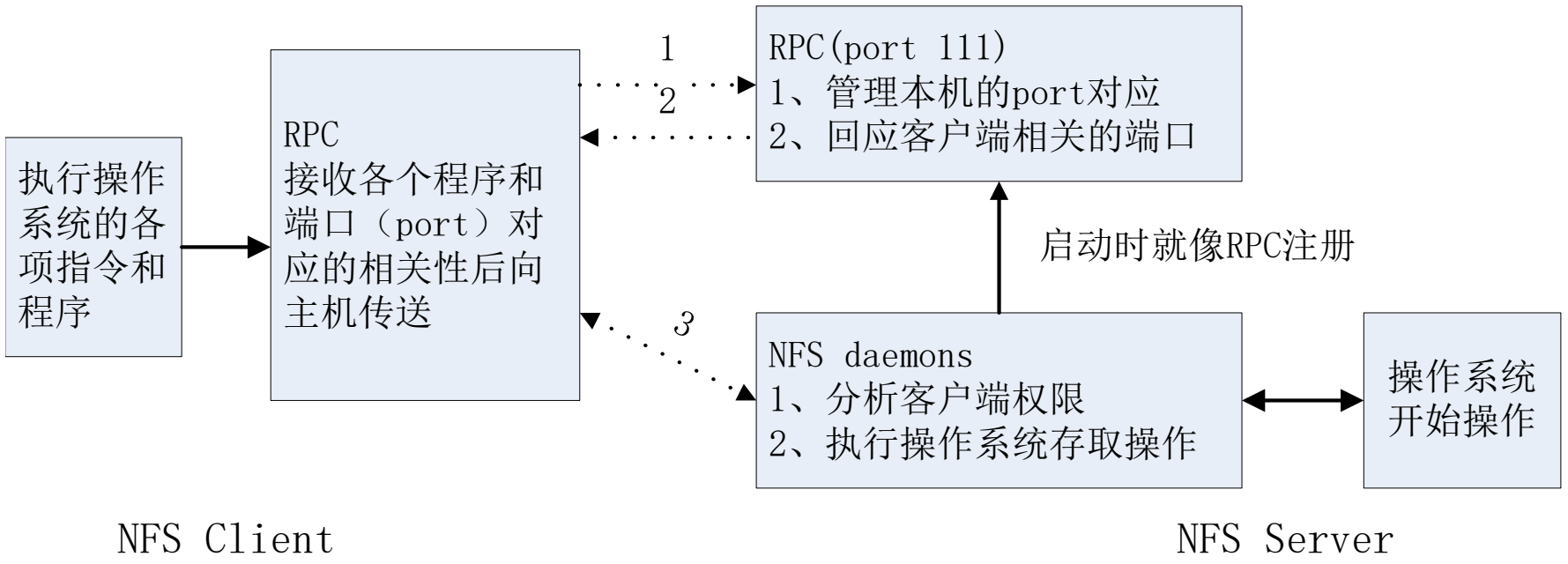
提纲

- 嵌入式文件系统简介
- Linux文件系统框架
- 本地文件系统
- 网络文件系统
- 根文件系统设计

NFS文件系统框架


NFS Client
NFS Server
系统调用（System Calls）
系统调用（System Calls）
VFS层
VFS层
NFS文件系统
NFS Daemon
本地文件系统
RPC/XDR
RPC/XDR
磁盘
网络(TCP/UDP)

NFS的PRC机制


执行操作系统的各项指令和程序
RPC
接收各个程序和端口（port）对应的相关性后向主机传送
1
2
3
RPC(port 111)
1、管理本机的port对应
2、回应客户端相关的端口
启动时就像RPC注册
NFS daemons
1、分析客户端权限
2、执行操作系统存取操作
操作系统
开始操作
NFS Client
NFS Server

提纲

- 嵌入式文件系统简介
- Linux文件系统框架
- 本地文件系统
- 网络文件系统
- 根文件系统设计

常用嵌入式根文件系统

- 本地文件系统
- RAM disk
- NFS

RAM disk的BootLoader配置

```
#setenv bootargs initrd=0x81000000,0x400000 root=/dev/ram rw
console=ttyS2,115200n8 mem=64M init=linuxrc
```

NFS的BootLoader配置

```
#setenv bootargs root=/dev/nfs rw
nfsroot=10.0.0.1:/share/rootfs
ip=10.0.0.2:10.0.0.1::255.255.255.0 console=ttyS2,115200n8
mem=64M
```

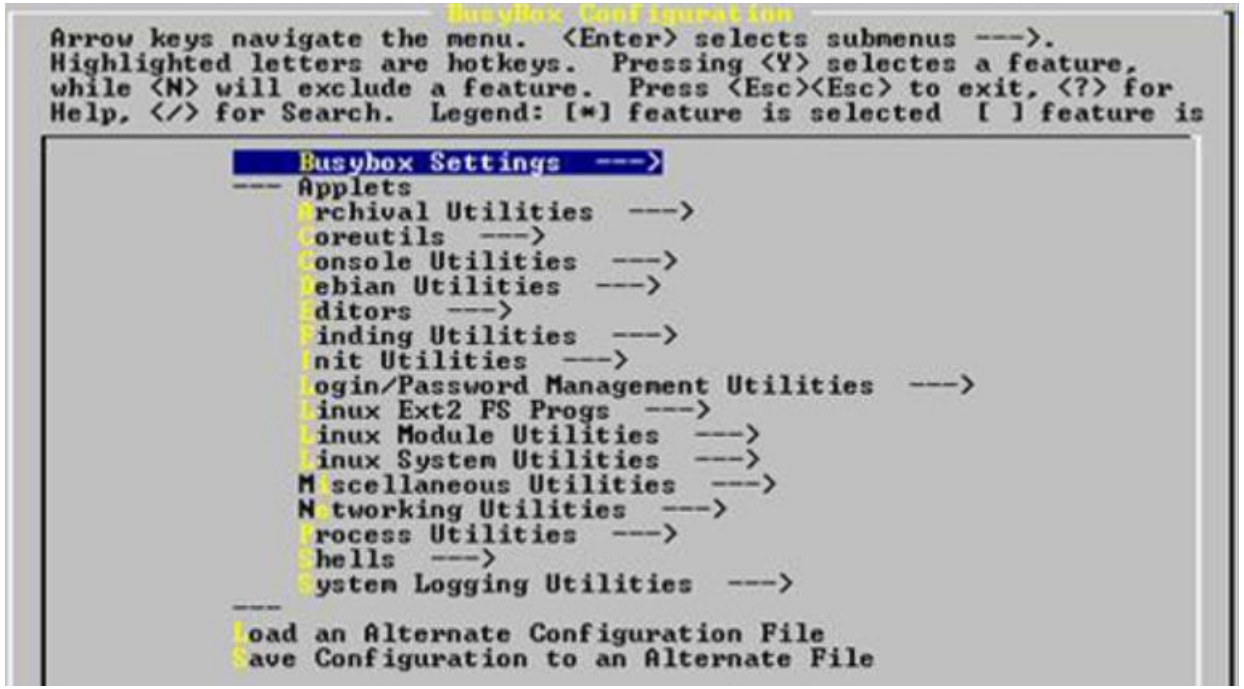
Busybox（1）

- Busybox 是 Debian GNU/Linux 的大名鼎鼎的 Bruce Perens首先开发。后来有许多 Debian developers贡献力量，这其中尤推 busybox 维护者 Erik Andersen，身患癌症，却是一名优秀的自由软件开发者

- Busybox包括一个迷你的vi编辑器，系统不可或缺的 /sbin/init程序，以及其他诸如 sed, ifconfig, halt, reboot, mkdir, mount, ln, ls, echo, cat ... 等等，大小也不过 100K 左右。用户还可根据自己的需要，决定到底要在busybox中编译进哪几个应用程序的功能，busybox的体积可以进一步缩小。

Busybox（2）

- Busybox支持多种体系结构，可以静态或动态链接glic或者uclibc库，以满足不同的需要，也可以修改Busybox默认的编译配置以移除不想使用的命令的支持



□谢　谢！