第二章

文件系统和标准I/O

讲师：任继梅
QQ:　59189174

课前提问

1. C库操作文件的函数有哪些?
2. 使用C库函数操作文件的基本过程是?
3. 文件分为哪两类?
4. Linux文件系统支持的文件类型有哪些?

本章内容

2.1系统调用

2.2 文件类型

2.3 文件系统

2.4 标准I/O文件操作

本章目标

- ✓ 掌握系统调用 ★★★
- ✓ 了解文件类型 ★★☆
- ✓ 掌握文件系统 ★★★
- ✓ 了解标准I/O文件操作 ★★☆

第一节

系统调用概述

知识点1-系统调用

C语言函数按提供源分类

- C语言标准库函数
- 操作系统提供的函数——系统调用
- 自定义函数
- 第三方函数

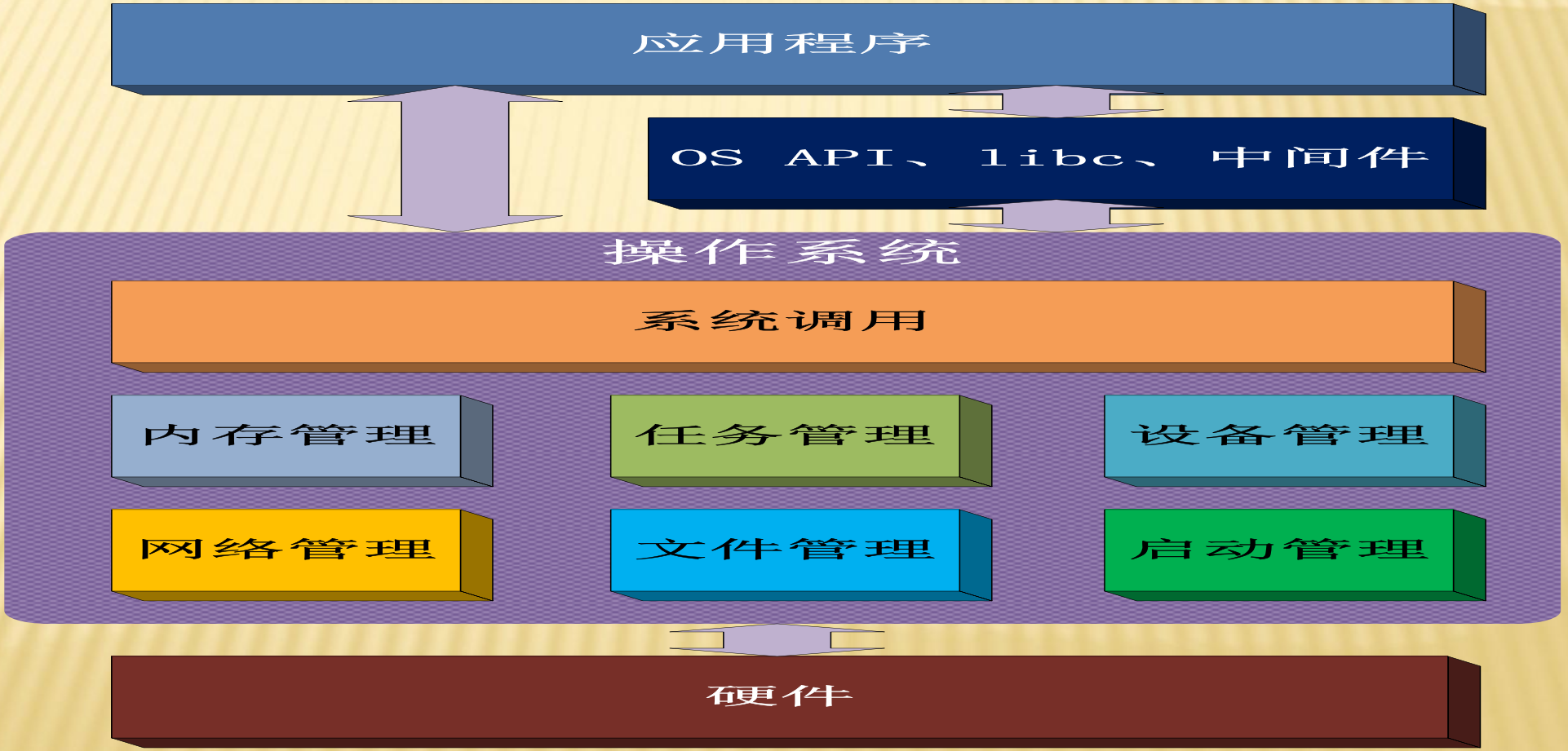
系统调用

Linux系统调用

- 准确称谓：由操作系统实现提供的API,即OS API，习惯上称为系统调用
- 是应用程序和操作系统之间的接口
- 包括常用系统调用和由系统调用派生出的函数
- 系统调用把应用程序的请求传给内核，调用相应的的内核函数完成所需的处理，将处理结果返回给应用程序
- 这些函数的实现代码位于操作系统内核

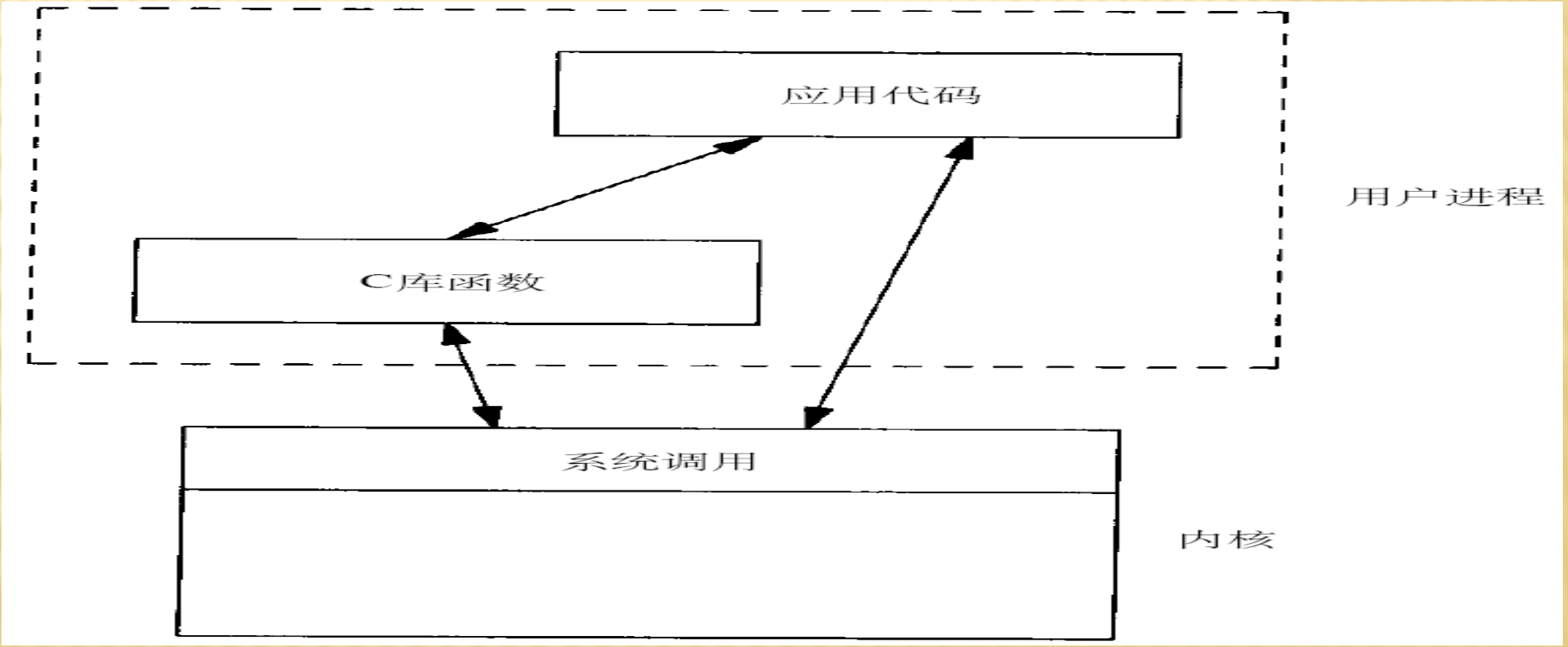
系统调用

- 计算机系统层次结构



系统调用

- Linux系统调用与C库



系统调用

Linux系统调用按功能分类

- 进程控制
- 文件系统控制
- 系统控制（如reboot、time等等）
- 内存管理（如brk、mmap等等）
- 网络管理
- 用户管理
- 进程间通信

第二节

文件类型

知识点2-文件类型

Unix家族的操作系统典型共同特点——一切皆文件

文件类型

- ls –l标注的文件类型
  - 普通文件
  - 目录文件
  - 链接文件
  - 管道文件
  - 设备文件
  - 套接字

文件类型

- 普通文件（regular file）
  - 驻留在磁盘中，包含自身数据的文件
  - 按能否使用文本查看工具直接阅读分为：
  - 文本文件
  - 二进制文件：所有非文本文件的统称
    - 如常见jpg、mp3、rmvb等等

```
[alex@alex ~]$ls -l minicom.log

-rw-rw-r-- 1 alex alex 2903829 Apr 23 22:41 minicom.log
```

文件类型

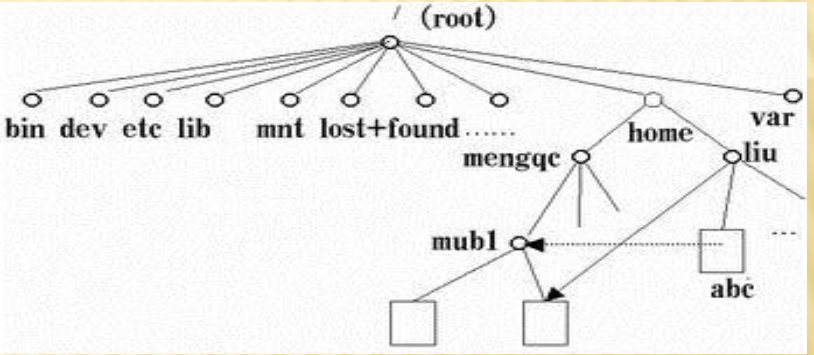
目录文件（directory）

- 管理和组织其他文件

```
[alex@alex~/spec]$ls -l

total 8784

- rw-r--r-- 1 alex alex 8970378 Dec 13 12:56 2800000 1.pdf

drwxrwxr-x 2 alex alex 4096 Nov 23 09:13 uart
```

文件类型

链接文件（link）



- 为一个文件起别名
- 分类
- 软链接（符号链接）
  - 一个小文本文件
  - 包含要链接的目标文件的绝对路径名
  - ln –s 源文件 软链接文件
- 硬链接
  - 一个文件多个名字
  - ln增加文件的链接数，rm减少文件的链接数
  - 链接数为0时彻底删除
  - 不能对目录操作
  - 同一文件系统

文件类型

管道文件（named pipe）

- 一种用于进程间传输数据的工具
- 详细见后续的进程间通信课程

```
[alex@alex ~]$mknod mypipe p

[alex@alex ~]$ls -l mypipe

prw-rw-r-- 1 alex alex 0 Apr 24 09:14 mypipe
```

文件类型

设备文件（device）

- Linux中每一个支持的I/O设备在/dev目录下都有一个对应的设备文件
- 从这样的设备文件里读数据实际就是从对应设备接收数据
- 向这样的设备文件里写数据实际就是向对应设备发送数据
- 因此，Linux中操作设备就像操作普通文件一样，应用程序无须关心操作硬件的具体细节
- 分类：
- 字符设备文件：无缓冲，顺序读写，如：键盘，鼠标
- 块设备文件：有缓冲，随机读写，如：硬盘

```
[alex@alex/dev]$ls -la tty

crw-rw-rw- 1 root root  5, 0  Apr 23 19:59 tty
```

```
[alex@alex/dev]$ls -la sda1

brw-r----- 1 root disk 8, 1 Apr 23 17:31 sda1
```

文件类型

- 套接字（socket）
  - 类似于管道文件
  - 用于不同计算机进程之间的网络通讯
  - 详细见后续《网络编程》课程

```
[alex@alex/tmp]$ls -l

srwxrwxr-x 1 alex alex  0 Apr 21 16:40 mapping-alex

srwxr-xr-x 1 root root  0 Mar 10 16:19 mapping-root
```

文件系统

与文件相关的数据

- 文件名
- 文件内容数据
- 文件属性信息（如创建者、访问权限、文件大小等等），也称为文件的元信息

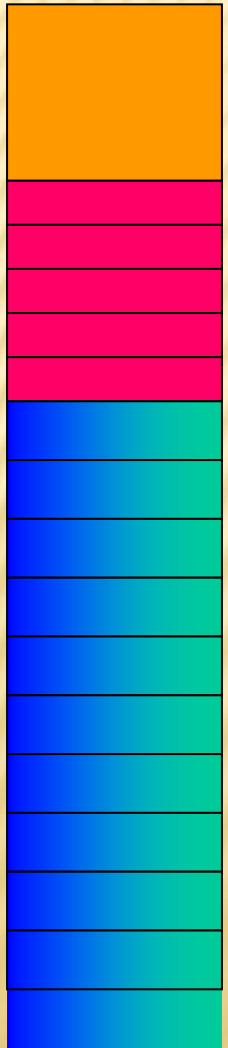
文件系统

硬盘上的数据组织

- 扇区：
  - 对硬盘进行存取的最小存储单位为扇区（Sector），每个扇区512字节
  - 操作系统读取硬盘的时候，不会一个个扇区地读取，这样效率太低
  - 而是一次性连续读取多个扇区，即一次性读取一个"块"（block）
- 块：
  - 这种由多个扇区组成的"块"，是文件存取的最小单位。
  - “块”的大小由文件系统决定，最常见的是1KB、2KB、4KB
  - 又称为数据块或逻辑块
- 分区：
  - 把空间很大的硬盘划分成几个相对独立的存储区域
  - 每个分区有且仅有一个文件系统，给分区指定文件系统的过程叫格式化
  - 新硬盘必须分区、格式化后才能使用

Linux文件系统结构

- 文件系统的三个区域



超级块 → 存放文件系统本身的结构信息

i-节点表 → 存放i-节点信息列表

属性

数据区 → 存放文件内容

文件系统

Extfs相关的几个名词

- 自举块：即引导块，主分区仅有一块
- 超级块：存放文件系统本身的信息，
- 每分区一个，包含块的大小，每个块组中包含的块数等等信息
- 块组描述符：本组的相关信息
- 块位图：每一位指出块组中对应的那个数据块是否被使用
- inode：储存文件元信息的块，中译名“索引节点”
- inode位图：每一位分别指出块组中对应的那个inode是否被使用
- inode表：存放所有文件的元数据（metadata）
- 数据块：存放普通文件内容

文件系统

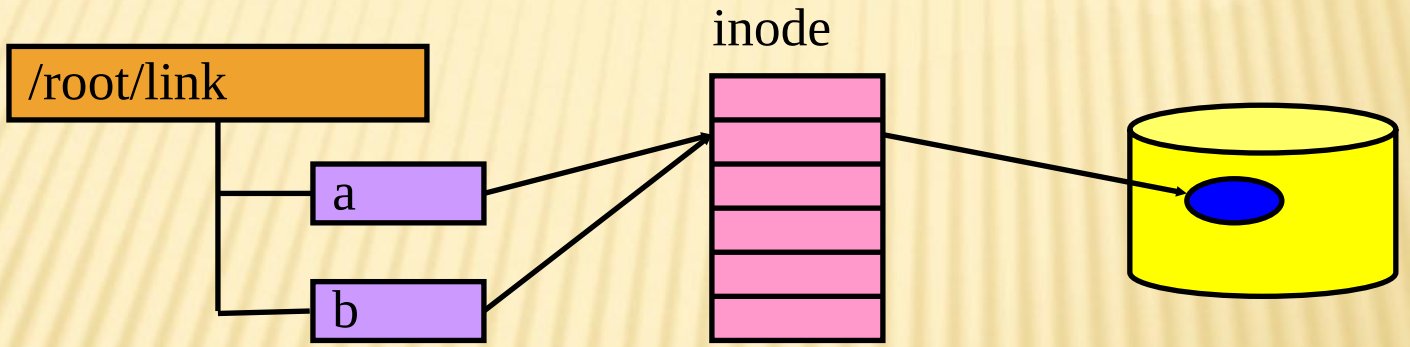
- 目录块：存放目录文件的内容
  - 根目录总是在inode表的第二项
  - 一对对的文件名（或子目录名）和inode编号

一个奇怪的问题

- 分区有足够多的空余但是就是无法创建新文件？？？

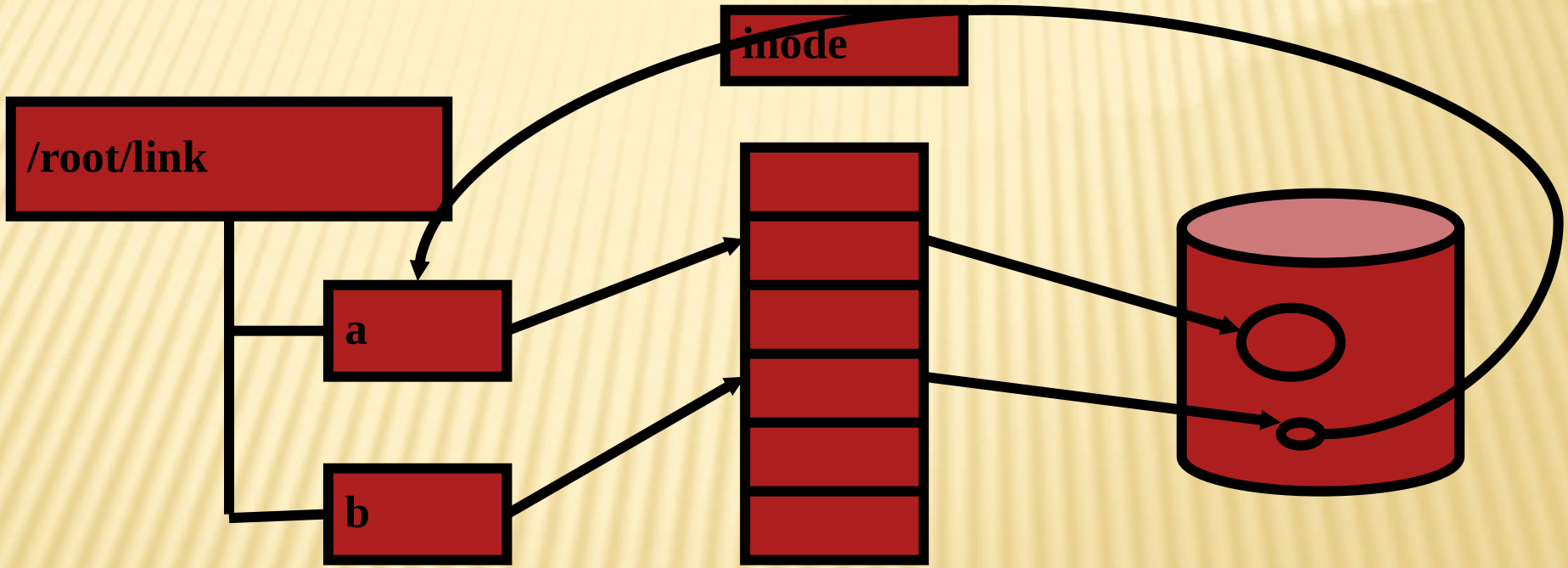
文件系统

- 硬链接



文件系统

- 再看软链接



第三节

标准I/O函数

知识点3-标准文件I/O操作

- **定义：**
  - 文件：一组相关数据的有序集合。
  - 文件名：这个数据集合的名称。
- **按类型分类：**
  - 常规文件
    - ASCII码文件
    - 二进制的文件

标准I/O库

- 不仅在UNIX系统，在很多操作系统上都实现了标准I/O库
- 标准I/O库由ANSI C标准说明
- 标准I/O库处理很多细节，如缓存分配、以优化长度执行I/O等，这样使用户不必关心如何选择合适的块长度
- 标准I/O在系统调用函数基础上构造的，它便于用户使用
- 标准I/O库及其头文件stdio.h为底层I/O系统调用提供了一个通用的接口。

标准I/O库

- 文件指针
  - FILE指针：每个被使用的文件都在内存中开辟一个区域，用来存放文件的有关信息，这些信息是保存在一个结构体类型的变量中，该结构体类型是由系统定义的，取名为FILE。
  - 标准I/O库的所有操作都是围绕流(stream)来进行的，在标准I/O中，流用FILE *来描述。
  - 标准I/O库是由Dennis Ritchie在1975年左右编写的
- 流（stream）
  - 定义：所有的I/O操作仅是简单的从程序移进或者移出，这种字节流，就称为流。
  - 分类：文本流/二进制流。

缓冲区

- **缓冲文件系统(高级磁盘IO)**
  - 目的：尽量减少使用read/write的调用
  - 定义：系统自动的在内存中为每一个正在使用的文件开辟一个缓冲区，从内存向磁盘输出数据必须先送到内存缓冲区，装满缓冲区在一起送到磁盘中去。从磁盘中读数据，则一次从磁盘文件将一批数据读入到内存缓冲区中，然后再从缓冲区逐个的将数据送到程序的数据区。
  - 分类：全缓存，行缓存，不缓存。
- **非缓冲文件系统 （低级磁盘IO）**
  - 定义：依靠于操作系统，通过操作系统的功能对文件进行读写，是系统级的输入输出。

缓冲区

- 标准I/O提供了三种类型的缓存
  - 全缓存
    - 当填满I/O缓存后才进行实际I/O操作，或者满足一定条件后，系统通过调用malloc来获得所需要的缓冲区域，默认值。
    - 刷新(fflush)：标准I/O的写操作。当缓冲区满了，或者满足一定的条件后，就会执行刷新操作。
  - 行缓存
    - 当在输入和输出中遇到新行符('\n')时，进行I/O操作。
    - 当流遇到一个终端时，典型的行缓存。
  - 不带缓存
    - 标准I/O库不对字符进行缓冲，例如stderr。
    - 很多的人机交互界面要求不可全缓存。
    - 标准出错决不会是全缓存的。
- 使用setbuf()和setvbuf()可以更改缓存的类型
- 在任何时刻，可以使用fflush强制刷新一个数据流

缓冲区操作

- **定义：**

```
#include <stdio.h>
int fflush(FILE *fp);
```

- **说明：**
  - 可强制刷新一个流。此函数使该流所有未写的数据都被传递至内核。

标准文件I/O操作

- 标准I/O预定义3个流，他们可以自动地为进程所使用

标准输入	0	STDIN_FILENO	stdin
标准输出	1	STDOUT_FILENO	stdout
标准错误输出	2	STDERR_FILENO	stderr

知识点4-I/O模型比较

I/O模型	文件I/O	标准I/O
缓冲方式	非缓冲I/O	缓冲I/O
操作对象	文件描述符	流(FILE *)
打开	open()	fopen()/freopen()/fdopen()
读	read()	fread()/fgetc()/fgets()...
写	write()	fwrite()/fputc()/fputs()...
定位	lseek()	fseek()/ftell()/rewind()/fsetpos()/fgetpos()
关闭	close()	fclose()

课程总结

本节课程内容

- 系统调用
- 文件类型
- 文件系统
- 标准文件I/O

下节课程

- 文件I/O
- 文件读写函数
- 目录操作
- 其他文件操作函数

QQ：　59189174
E-mail：yumeifly@sohu.com