目录

[Linux系统 1](#_Toc514423875)

[Linux设计目标 1](#_Toc514423876)

[Shell 1](#_Toc514423877)

[cp和mv 1](#_Toc514423878)

[chmod 1](#_Toc514423879)

[head 2](#_Toc514423880)

[make 2](#_Toc514423881)

[mkdir 2](#_Toc514423882)

[od 2](#_Toc514423883)

[ps 2](#_Toc514423884)

[rmdir 2](#_Toc514423885)

[tail 2](#_Toc514423886)

[内核结构 2](#_Toc514423887)

[内核作用 2](#_Toc514423888)

[内核五个子系统 3](#_Toc514423889)

[Linux shell 常用命令 3](#_Toc514423890)

# Linux系统

## Linux设计目标

能够同时处理多进程和多用户的交互式系统。

## Shell

### cp和mv

cp a b 把文件a移动到b，复制

mv a b 把文件a移动到b，删除a。

### chmod

修改文件保护模式

### head

提取文件前几行.

### make

编译文件生成二进制文件

### mkdir

创建目录

### od

以八进制显示一个文件

### ps

列出正在运行的进程

### rmdir

删除一个目录

### tail

提取文件最后几行

## 内核结构

### 内核作用

内核在硬件的上一层，负责I/O设备与存储管理单元的交互，并控制CPU对设备的访问。为应用程序提供接口。核心，管理硬件设备，供应用程序使用。

### 内核五个子系统

1. 进程管理和调度：负责管理CPU资源，使得各个进程可以公平的访问CPU。
2. 内存管理：管理内存资源，使得各个进程可以安全地共享机器的内存资源。
3. VFS（虚拟文件系统）：体现了Linux系统的“一切皆是文件”。将不同功能的外部设备，抽象为可以通过统一的文件操作接口。
4. Network（网络子系统）：负责管理系统的网络设备
5. IPC（进程间通信）：IPC不管理任何的硬件，主要负责Linux系统中进程之间的通信。

#### 进程管理和调度的四个模块

1. Scheduling Policy：实现进程调度的策略，它决定哪个（或哪几个）进程将拥有CPU。
2. Architecture-specific Schedulers:把对cpu的控制抽象为统一的接口，。
3. Architecture-independent Scheduler:和第一个模块沟通，决定接下来执行哪个进程
4. System Call Interface:系统调用接口

#### 内存管理

主要是硬件内存和虚拟内存之间的映射

#### 虚拟文件系统（VFS）

管理各种各样不同的文件系统（FAT/FAT32/NTFS），屏蔽它们的差异，以统一的方式，为用户程序提供访问文件的接口。

#### 网络子系统

管理各种网络设备，实现各种网络协议栈，最终实现通过网络连接管理其它系统的功能。

## Linux shell 常用命令

<https://www.cnblogs.com/zhangziqiang/p/7478075.html>