21世纪高等学校规划教材 | 计算机应用

基于UNIX/Linux的C系统编程

张杰敏 王巍 编著

清华大学出版社

基于UNIX/Linux的C系统编程

张杰敏 王巍 编著

清华大学出版社
北京

内 容 简 介

本书面向应用组织内容，由浅入深地对 UNIX/Linux 环境下的系统编程进行全面分析，注重系统编程思想和系统编程模式的论述。全书共分为 7 章，详尽而细致地介绍了系统编程的概念及基础知识、文件操作、基于进程的并发控制技术、基于线程的并发控制技术、网络通信技术、异步事件编程和并行编程等多方面的内容。书中各章设计和选用了大量实例，以“案例+编程模式→原理+例程→系统编程思想”为体系，使读者易于理解和应用，同时也为读者的拓展和创新留有空间。

本书适用于计算机及相关专业，也可供系统编程人员和工程技术人员参阅。使用本书需要具备程序设计基础，了解操作系统原理。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。
版权所有，侵权必究。侵权举报电话：010-62782989　13701121933

图书在版编目（CIP）数据

基于 UNIX/Linux 的 C 系统编程 / 张杰敏，王巍编著. --北京：清华大学出版社，2013.4
21 世纪高等学校规划教材 • 计算机应用
ISBN 978-7-302-31283-3

Ⅰ. ①基… Ⅱ. ①张… ②王… Ⅲ. ①C 语言-程序设计-高等学校-教材 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2013）第 006888 号

责任编辑：闫红梅　李　晔
封面设计：傅瑞学
责任校对：时翠兰
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn,010-62795954
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：19.25　　**字　　数**：469 千字
版　　次：2013 年 4 月第 1 版　　**印　　次**：2013 年 4 月第1 次印刷
印　　数：1～3000
定　　价：34.50 元

产品编号：047276-01

编审委员会成员

（按地区排序）

清华大学	周立柱　教授
	覃　征　教授
	王建民　教授
	冯建华　教授
	刘　强　副教授
北京大学	杨冬青　教授
	陈　钟　教授
	陈立军　副教授
北京航空航天大学	马殿富　教授
	吴超英　副教授
	姚淑珍　教授
中国人民大学	王　珊　教授
	孟小峰　教授
	陈　红　教授
北京师范大学	周明全　教授
北京交通大学	阮秋琦　教授
	赵　宏　副教授
北京信息工程学院	孟庆昌　教授
北京科技大学	杨炳儒　教授
石油大学	陈　明　教授
天津大学	艾德才　教授
复旦大学	吴立德　教授
	吴百锋　教授
	杨卫东　副教授
同济大学	苗夺谦　教授
	徐　安　教授
华东理工大学	邵志清　教授
华东师范大学	杨宗源　教授
	应吉康　教授
东华大学	乐嘉锦　教授
	孙　莉　副教授
浙江大学	吴朝晖　教授
	李善平　教授

扬州大学	李　云　教授
南京大学	骆　斌　教授
	黄　强　副教授
南京航空航天大学	黄志球　教授
	秦小麟　教授
南京理工大学	张功萱　教授
南京邮电学院	朱秀昌　教授
苏州大学	王宜怀　教授
	陈建明　副教授
江苏大学	鲍可进　教授
中国矿业大学	张　艳　教授
武汉大学	何炎祥　教授
华中科技大学	刘乐善　教授
中南财经政法大学	刘腾红　教授
华中师范大学	叶俊民　教授
	郑世珏　教授
	陈　利　教授
江汉大学	颜　彬　教授
国防科技大学	赵克佳　教授
	邹北骥　教授
中南大学	刘卫国　教授
湖南大学	林亚平　教授
西安交通大学	沈钧毅　教授
	齐　勇　教授
长安大学	巨永锋　教授
哈尔滨工业大学	郭茂祖　教授
吉林大学	徐一平　教授
	毕　强　教授
山东大学	孟祥旭　教授
	郝兴伟　教授
厦门大学	冯少荣　教授
厦门大学嘉庚学院	张思民　教授
云南大学	刘惟一　教授
电子科技大学	刘乃琦　教授
	罗　蕾　教授
成都理工大学	蔡　淮　教授
	于　春　副教授
西南交通大学	曾华燊　教授

此为试读，需要完整PDF请访问：www.ertongbook.com

出版说明

随着我国改革开放的进一步深化，高等教育也得到了快速发展，各地高校紧密结合地方经济建设发展需要，科学运用市场调节机制，加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度，通过教育改革合理调整和配置了教育资源，优化了传统学科专业，积极为地方经济建设输送人才，为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是，高等教育质量还需要进一步提高以适应经济社会发展的需要，不少高校的专业设置和结构不尽合理，教师队伍整体素质亟待提高，人才培养模式、教学内容和方法需要进一步转变，学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007 年 1 月，教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》，计划实施“高等学校本科教学质量与教学改革工程（简称‘质量工程’）”，通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容，进一步深化高等学校教学改革，提高人才培养的能力和水平，更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中，各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势，对其特色专业及特色课程（群）加以规划、整理和总结，更新教学内容、改革课程体系，建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上，经教育部相关教学指导委员会专家的指导和建议，清华大学出版社在多个领域精选各高校的特色课程，分别规划出版系列教材，以配合“质量工程”的实施，满足各高校教学质量和教学改革的需要。

为了深入贯彻落实教育部《关于加强高等学校本科教学工作，提高教学质量的若干意见》精神，紧密配合教育部已经启动的“高等学校教学质量与教学改革工程精品课程建设工作”，在有关专家、教授的倡议和有关部门的大力支持下，我们组织并成立了“清华大学出版社教材编审委员会”（以下简称“编委会”），旨在配合教育部制定精品课程教材的出版规划，讨论并实施精品课程教材的编写与出版工作。“编委会”成员皆来自全国各类高等学校教学与科研第一线的骨干教师，其中许多教师为各校相关院、系主管教学的院长或系主任。

按照教育部的要求，“编委会”一致认为，精品课程的建设工作从开始就要坚持高标准、严要求，处于一个比较高的起点上；精品课程教材应该能够反映各高校教学改革与课程建设的需要，要有特色风格、有创新性（新体系、新内容、新手段、新思路，教材的内容体系有较高的科学创新、技术创新和理念创新的含量）、先进性（对原有的学科体系有实质性的改革和发展，顺应并符合 21 世纪教学发展的规律，代表并引领课程发展的趋势和方向）、示范性（教材所体现的课程体系具有较广泛的辐射性和示范性）和一定的前瞻性。教材由个人申报或各校推荐（通过所在高校的“编委会”成员推荐），经“编委会”认真评审，最后由清华大学出版社审定出版。

目前，针对计算机类和电子信息类相关专业成立了两个“编委会”，即“清华大学出版社计算机教材编审委员会”和“清华大学出版社电子信息教材编审委员会”。推出的特色精品教材包括：

（1）21 世纪高等学校规划教材·计算机应用——高等学校各类专业，特别是非计算机专业的计算机应用类教材。

（2）21 世纪高等学校规划教材·计算机科学与技术——高等学校计算机相关专业的教材。

（3）21 世纪高等学校规划教材·电子信息——高等学校电子信息相关专业的教材。

（4）21 世纪高等学校规划教材·软件工程——高等学校软件工程相关专业的教材。

（5）21 世纪高等学校规划教材·信息管理与信息系统。

（6）21 世纪高等学校规划教材·财经管理与应用。

（7）21 世纪高等学校规划教材·电子商务。

（8）21 世纪高等学校规划教材·物联网。

清华大学出版社经过三十多年的努力，在教材尤其是计算机和电子信息类专业教材出版方面树立了权威品牌，为我国的高等教育事业做出了重要贡献。清华版教材形成了技术准确、内容严谨的独特风格，这种风格将延续并反映在特色精品教材的建设中。

清华大学出版社教材编审委员会
联系人：魏江江
E-mail:weijj@tup.tsinghua.edu.cn

前言

随着知识经济在国民经济结构中所占比重的逐步扩大，软件产业作为国家战略性先导产业，已越来越为人们所关注。尽管该产业前景诱人且人才缺口较大，但从事软件行业并期望具备持续发展的能力则并非易事。软件开发技术更新过快、人员淘汰率过高，是当前软件业的主要行业特色。因此，学习编程最重要的是理解程序设计思想，掌握程序设计模式，形成良好的程序设计习惯。变化中的各种应用问题，都是思想和模式的体现。理解了编程思想、掌握了编程模式，才能有效适应软件技术的发展。有了良好的程序设计习惯，才有助于编写可读性好的程序代码。

从事软件开发的程序员通常有两种技术路线可供选择：

- 应用级编程。
- 系统级编程。

应用级编程接近于计算问题本身，程序员更多关注的是程序目标，而不必追究硬件或底层的实现细节。系统级编程则更接近硬件，程序员必须熟悉操作系统环境或相关硬件的驱动、管理方式。应用级编程是系统级编程的学习基础，但就编程思想和书写规范而言，系统级编程与应用级编程没有本质区别。

本套书由上、下两册组成，上册《基于 UNIX/Linux 的 C 程序设计》定位于应用级编程；下册《基于 UNIX/Linux 的 C 系统编程》定位于系统级编程。上、下册秉承一致的理念，围绕应用展开，注重编程思想和编程模式的论述，脱离繁杂的语法细节，使语言工具化。同时，在程序设计中展示良好的设计习惯，实现高质量的程序代码。

《基于 UNIX/Linux 的 C 系统编程》通过将相对稳定的操作系统 UNIX 和相对稳定的开发工具 C 语言相结合，总结并提出一系列系统编程模式以供读者学习和使用，使读者具备编写系统级程序的能力，同时，对 UNIX 系统的理解也将更加深入。

本书面向应用组织内容，这有别于传统组织方式。面向应用，对于系统级编程需要解决的问题更具针对性，也更能体现系统编程思想。对于读者来说，是一种更容易掌握系统编程模式的组织方式。

系统级编程的书写格式和编程思想与应用级编程相同，只是加入了大量的系统调用和特殊应用处理。系统级编程的操作对象为文件或进程。文件是程序或数据在外存中的组织形式，进程是程序或数据在内存中的组织形式。本书据此将内容分为两大部分，分别是基于文件的系统编程和基于进程的系统编程。

本书第 1 章，在应用级编程（《基于 UNIX/Linux 的 C 程序设计》）基础上，引申系统级编程的概念，介绍系统调用的基础知识。相对于应用级编程，系统级编程更贴近硬件，是面向底层操作的编程。

在 UNIX 概念中，磁盘文件、目录和设备都视为文件。第 2 章介绍文件操作，包括 UNIX

文件和文件系统的基本知识，UNIX 文件系统调用，目录和设备文件的系统调用等。用统一的文件操作形式，处理目录、设备等系统编程问题。

第 3～6 章介绍进程操作。多进程环境下，需要解决进程间的并发、异步和通信问题。第 3 章介绍多进程并发，包括 UNIX 进程的基本知识，进程控制，进程同步，以及进程间通信等。第 5 章介绍进程间的网络通信，包括 socket 的基本知识，面向连接的通信，面向无连接的通信，基于 IP 层和数据链路层的通信，以及 socket 并发编程模型等。第 6 章介绍多进程异步，包括异步编程的基本知识，信号系统的调用，时间的获取与设置，定时器的使用，以及异步 I/O 编程等。由于线程为轻量级进程，针对线程的系统级编程存在特殊性，第 4 章讨论多线程并发，包括 UNIX 线程的基本知识，线程控制和线程同步。

随着多核系统、网络、云计算概念的发展普及，相对于并发，并行编程愈发重要。第 7 章以系统编程的概念诠释并行编程的基本思想，包括串行计算与并行计算的基本知识，并行计算与云计算的关系，多机环境下的多进程并行编程，多核环境下的多线程并行编程，以及多机多核环境下的混合编程等。

本书设计的案例针对现实问题，具有实用价值，可运行并附有的具备说服力的运行结果。每个案例并不是孤立存在的，部分案例横向间存在对比说明的关系，案例纵向间存在由浅入深、层层递进的关系，直至达到一定的深度。例如，有关网络编程的案例已涉及 IP 层或数据链路层，这在国内外相关图书中少见叙及。通过这些案例展示了 C 语言可达到的应用深度和广度，同时也印证 UNIX 操作系统的设计思想。

全书的内容及案例的设计，都始终围绕着操作系统的应用展开，注重的是体现系统编程思想，介绍系统编程模式。换句话说，本书避免对系统调用函数做语法的解析或使用方式的拆解，而是将系统调用函数置于实际应用的上下文环境，通过系统编程模式体现系统调用函数的功能。

系统编程模式是一个完整的思想体系，是系统编程的灵魂，蕴含在千变万化的应用中。理解了系统编程思想、掌握了系统编程模式，针对任何实际问题拥有系统调用函数的手册就足够了，就像查字典一样。这是追求的目标，也是本书的特点。

注重系统编程模式，使许多概念有了新的诠释方式。例如，一般将文件描述为“字节的序列”，这种抽象的描述无助于对实际问题的理解和解决。本书将磁盘文件映射为 UNIX 操作文件，通过介绍映射过程帮助读者建立文件的概念，使得文件不再抽象，而落实为系统编程的操作对象。

再如，异步编程是一种事件驱动的编程模式，主要通过信号技术予以实现。将信号置于异步的上下文环境，既体现了其在系统编程中的作用，又简化了操作过程的烦琐性。

书中的程序代码承接《基于 UNIX/Linux 的 C 程序设计》的程序设计思想，可读性好，示范了良好的编程习惯，且可扩展。为读者实现自己的编程思想留有空间。

本书由张杰敏、王巍编著。张杰敏编写了第 1 章和第 2 章。王巍编写了第 3～7 章，并编写了各章的习题。张杰敏审阅了全书。在此对付出心血的各位同仁表示诚挚敬意和感谢。

由于作者水平有限，书中难免存在错误和不妥之处，恳请读者批评指正。编者的联系方式为：together_with_you@126.com。

编者

2012 年 12 月

目 录

此为试读，需要完整PDF请访问：www.ertongbook.com

第1章 基础知识

UNIX 自 1969 年诞生以来，已经发展为 System III & V、BSD 和 Linux 三大分支。其中，Linux 由于其具有的独特设计理念和开源模式，已成为当前 UNIX 应用的主流和学习、研究操作系统内部工作细节的主要平台，所以从现在起，在书中只要提及 UNIX 一词，其概念均包含 Linux，除非另有说明。

本章将学习系统级编程的基础知识，了解系统编程的概念、作用和实现方式，并且简要介绍 UNIX 内核的体系结构及其提供的各种功能，为以后章节学习系统编程提供理论基础和入门介绍。

通过对本章的学习，应该掌握如下知识：

- 什么是系统编程。
- 系统编程能做什么。
- 系统编程如何实现。
- 自己动手编写一个系统程序。
- 库的使用。
- 如何学习系统编程。

1.1 什么是系统编程

许多读者对编程都有所了解，也可能写过各种各样的程序，例如：

- 使用 JSP 语言和 SQL 数据库，开发一套用于新闻发布的网站系统。
- 使用 3DMax 脚本语言，编写一套能够交互的动漫游戏。
- 使用 VBA 编程，实现 Excel 各工作表数据的快速汇总。
- 使用 C 语言，编写一个通过积分法求解 PI 值的程序。
- 使用 Java 语言和 Oracle 数据库，开发一套企业进销存管理信息系统等。

上述程序均运行于操作系统层之上的软件应用层，常被称为应用级编程。应用级编程广泛存在，大多数软件都使用此类编程方式实现，它主要用于解决各领域用户的应用需求。

由于远离操作系统内核，一些计算问题无法通过应用级编程解决，比如高性能计算领域、嵌入式开发领域、多机与多核环境下的并行计算领域、大规模游戏开发领域、网络安全领域和多媒体处理领域等。由于在这些应用领域的编程需要了解操作系统的运行机制，并熟悉底层程序的编写，所以此类应用下的编程，常被称为系统级编程，简称系统编程。

就编程思想和书写规范而言，系统级编程与应用级编程没有本质区别，主要依据基于以下两点进行划分：

- 系统编程更接近硬件，程序员必须熟悉操作系统环境或相关硬件的驱动、管理方式；而应用级编程则更接近于计算问题本身，其程序员更多关注的是程序目标，而不必追究硬件或底层的实现细节。
- 系统编程最主要的实现手段是系统调用。系统调用采用陷入方式（即软中断方式）实现目态（用户程序执行时的状态）到管态（又称核心态，操作系统程序执行时的状态）之间的切换；而应用级编程则止步于调用函数库，对于函数库以下的实现细节不做考虑。

总之，系统级编程是通过与操作系统的直接交互实现的，而应用级编程则运行在操作系统之上，其相对于操作系统的编程界面参见图 1.1。随着计算应用的深化，应用级编程有远离系统编程的趋势，系统级编程与应用级编程的作用愈加分化。

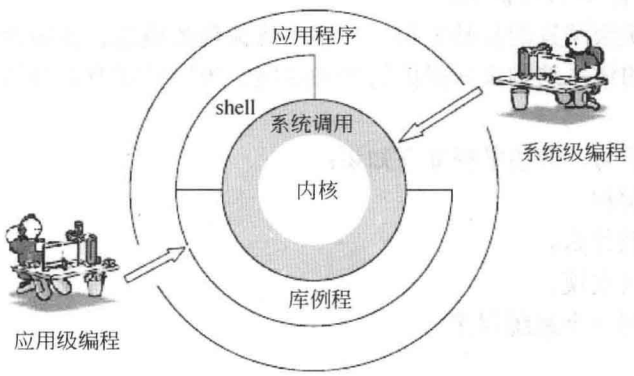

图 1.1 UNIX 的系统调用

1.1.1 系统调用

计算机系统的内存分为系统空间和用户空间。系统空间是指操作系统内核正常运行所需要占用的内存空间。系统空间之外的为用户空间，为用户进程提供必需的内存空间。这种内存管理机制影响着操作系统的调度和管理机制，是系统和用户程序能够正确运行的重要硬件保障。

通常，用户进程是不能够直接访问系统空间、调用内核函数的，但用户进程可以通过系统调用技术进入内核（系统空间），获取内核服务。所谓系统调用，是指一种进入系统空间的方法。UNIX 和 Linux 的系统调用均以函数形式存在，并以此为用户进程提供应用接口。不同的系统调用目的由不同的系统调用函数实现。用户进程对系统调用函数的每一次成功调用，都将进行一次相应的用户空间到系统空间的转换。进入内核后，不同的系统调用会找到各自对应的内核函数入口地址，由内核函数提供服务。

以调用 Linux 0.01 系统下 fork 函数为例来分析系统调用的实现过程。fork 函数用于创建一个新进程（将在第 3 章详细讨论），是典型的系统调用函数。

如图 1.2 所示，调用 fork 函数实际上是调用了中断 0x80，通过初始化的中断描述符表 IDT，程序转移到_system_call，最终通过一个函数指针数组 sys_call_table[]转化成了调用 sys_fork 函数。

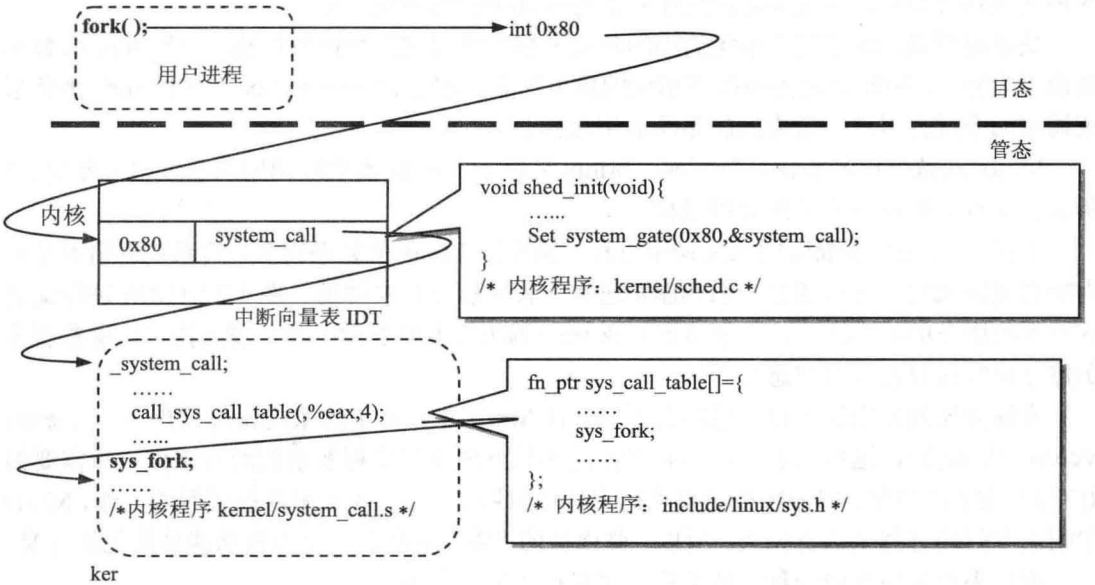

图 1.2　Linux 0.01 的 fork 函数系统调用

通过分析上述的调用过程可知，系统调用起到了连接用户进程和操作系统的桥梁作用，桥的两端分别是目态下的 fork 函数和管态下的 sys_fork 函数。由于桥的两端分别处于不同的特权级，所以需要使用中断来进行过渡。

各种不同的操作系统都有自己的系统调用，如 Windows API，便是 Windows 操作系统的系统调用。在 UNIX 规范中大约有 1108 个系统调用，在 Linux 规范中也有 260 个系统调用（定义在/include/linux/sys.h 中）。鉴于 Linux 内核代码完全公开，非常适合细致地分析系统调用的机制，所以本书的所有实验都是基于 Linux 平台的。

因此，在进行系统级编程时必须要依据不同内核的特点和要求，这有别于应用级编程。比如，在使用系统调用时，由于系统空间中缺少内存保护机制，且堆栈容量有限，所以其调用的嵌套层数不能过多，同时调度关系也必须要考虑内核执行路径的连续性，不能有长睡眠行为等。类似的限制还有很多，这些都是系统编程需要考虑的问题。

通过上述讨论，初步了解了系统编程。但在进行系统编程时，必须要对系统的结构和工作方式有更深入的了解，要知道什么是内核，内核能提供哪些服务（系统调用），如何使用它们，系统有哪些资源和设备，不同的资源和设备如何操作等。1.1.2 节将就内核、内核结构与内核服务展开讨论，后续章节逐步讨论其余问题。

1.1.2　内核与内核服务

内核是运行程序和管理各种硬件设备的核心程序，是操作系统的灵魂。操作系统内核

的设计理念或特性直接影响着操作系统的运行效率。依据内核的不同结构，现代操作系统可划分为两大类别，即微内核型操作系统和宏内核型操作系统。有关这两种内核的优劣之争始终不能停息，一直延续到今天，双方都无法说服对方。究其原因，主要是在理论上这两种内核各有长短，且在实际应用中也各有成功的操作系统出现。

受题材所限，这里仅从系统调用的角度考察并学习这两种内核结构，仍以 fork 函数系统调用为例。UNIX 是宏内核操作系统的典型代表，前面介绍的 Linux 下实现 fork 函数系统调用的全过程展现了宏内核的系统调用过程。

Minix 是微内核的操作系统实例，Minix 环境下 fork 函数系统调用的全过程（如图 1.3 所示）展现了微内核的系统调用过程。

由图 1.3 可知，Minix 的系统调用过程显然要比 Linux 的复杂得多，其系统调用不是由内核直接完成的，而是通过一系列服务进程（称为服务）实现的。图 1.3 中 MM 进程就是这一系列服务的其中之一，用来负责 fork 函数调用时所要做的工作。需要指出，这些服务仍然是运行在目态（用户态）。

实现方法的差别源于设计思路的不同。在 Minix 中，真正的系统调用只有 3 个（send、receive 和 both），这意味着内核不必事无巨细地处理用户进程要求的所有工作，只需要做好它的“邮局”职能，将消息按照要求往来传送就够了。在 Linux 中内核所做的工作，Minix 中则由专门的进程来负责完成。所以，微内核的“微”字指的是让内核功能最简化的意义。

通过系统调用比较两种内核结构，可以得出以下结论：

- 宏内核的优势在于其逻辑简单，直截了当，实现起来也容易。整个内核是一个不可分割的静态可执行体，并且必须以完整、单独的可执行块的形式在系统空间内运行。由于内核模块间的调用形式只有一种，即函数调用，除了函数调用所产生的开销外，没有额外开销，所以其内核的执行效率很高，可以支持庞大的操作系统中数以千计的函数调用。但是对于一些复杂的调用关系，由于其结构的特点使得其操作系统维护比较困难。具体实例包括 UNIX、Linux、MS-DOS、VMS、MVS、OS/360 和 MULTICS 等。
- 微内核的优势在于其结构严谨和精致，而且程序更容易模块化，进而更容易移植。但是内核与各个服务进程之间必须通过消息机制进行交互，内核发出请求，服务进程作出应答，这使得该结构下的操作系统通信开销过大，进而导致系统的整体执行效率大打折扣。具体实例有 RC4000、Amoeba、Chorus、Mach、Windows NT 等。

由于各种原因，Minix 并没有被广泛应用。相比之下，Linux 经过这些年的发展应用极为广泛，Linux 的成功并不简单地意味着宏内核结构优于微内核结构，因为对这两种操作系统的深入研究比盲目下结论更为重要。

在了解了内核结构后，讨论内核所提供的服务。

1．系统资源

内核服务是为了实现程序对系统资源的直接访问，系统资源包括以下内容。

1）处理器

众所周知，程序是由指令构成的，处理器是执行指令的硬件设备，一个系统中可能有多个处理器。内核能够安排一个程序何时开始执行，何时暂时停止、恢复执行以及何时终止执行。

此为试读，需要完整PDF请访问：www.ertongbook.com