李　杰　刘林阴　陈慧丽◎编著

嵌入式Linux系统开发原理与实战

QIANRUSHI LINUX XITONG
KAIFA YUANLI YU SHIZHAN

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

嵌入式 Linux 系统开发原理与实战

李　杰　刘林阴　陈慧丽　编著

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

内 容 简 介

本书介绍了嵌入式系统的基本概念和嵌入式系统的设计与开发原理及方法，内容涉及嵌入式系统软硬件的组成及特点、系统的设计原则和方法、嵌入式系统的开发工具、系统集成和测试方法，同时介绍了嵌入式系统的设计和开发实例，通过实战演练，详细具体地介绍了各种编程方法和编程技巧、开发工具的使用方法和使用技巧，以及硬件系统设计的详细过程和硬件系统的工作原理。本书适合作为电子类、计算机类、自动化类和机电类等专业的参考书，也可作为专业技术人员的培训参考资料。

图书在版编目（CIP）数据

嵌入式 Linux 系统开发原理与实战 / 李杰，刘林阴，陈慧丽编著. —北京：北京理工大学出版社，2017.11

ISBN 978-7-5682-4965-2

Ⅰ. ①嵌… Ⅱ. ①李… ②刘… ③陈… Ⅲ. ①Linux 操作系统–程序设计 Ⅳ. ①TP316.89

中国版本图书馆 CIP 数据核字（2017）第 275107 号

出版发行 / 北京理工大学出版社有限责任公司
社　　址 / 北京市海淀区中关村南大街 5 号
邮　　编 / 100081
电　　话 /（010）68914775（总编室）
　　（010）82562903（教材售后服务热线）
　　（010）68948351（其他图书服务热线）
网　　址 / http://www.bitpress.com.cn
经　　销 / 全国各地新华书店
印　　刷 /
开　　本 / 787 毫米×1092 毫米　1/16
印　　张 / 17.5
字　　数 / 413 千字
版　　次 / 2017 年 11 月第 1 版　2017 年 11 月第 1 次印刷
定　　价 / 58.00 元

责任编辑 / 钟　博
文案编辑 / 钟　博
责任校对 / 周瑞红
责任印制 / 施胜娟

前言 Preface

嵌入式计算的某些挑战在台式机计算世界中是众所周知的。例如，为从带流水线的高速缓存体系结构中获得最高性能，经常需要仔细分析程序轨迹。类似的，随着嵌入式系统的复杂性不断增加，在软件工程中针对特定复杂系统开发的技术变得十分重要。另外一个例子是设计多进程系统。对于台式机，通用操作系统的需求与实时操作系统的需求是截然不同的。过去 30 年针对大型实时系统开发的实时技术如今已普遍用于基于微处理器的嵌入式系统。嵌入式计算还面临一些新的挑战。一个较好的例子是功耗问题。在传统计算机系统中功耗已经不是一个主要考虑因素，但是对于用电池供电的嵌入式计算机，这是一个基本考虑因素，而且其在功耗容量受重量、成本或噪声等限制的情况下是十分重要的。另一个挑战是截止时限驱动的程序设计。嵌入式计算机常常对程序完成的期限作硬性限制。这种形式的限制在台式机世界里是罕见的。随着嵌入式处理器的速度越来越快，高速缓存和其他 CPU 单元也使执行时间越来越难以预测。然而，通过仔细分析和巧妙编程，可以设计可预测执行时间的嵌入式程序，甚至面对高速缓存等不可预测的系统部件也如此。

本书系统深入地介绍了嵌入式系统原理与设计方面的知识。本书从嵌入式系统的概念、应用领域、设计开发流程和发展趋势等基础知识出发，以 ARM 处理器为例，介绍了嵌入式处理器的知识及使用；以几种通用的嵌入式操作系统为例，着重介绍了嵌入式操作系统的基本知识，并根据在嵌入式硬件及软件设计方面的实际经验，图文并茂地介绍了嵌入式应用程序设计。

随着信息与通信技术的飞速发展，嵌入式系统技术在网络通信、电子消费、移动互联、工业控制等领域得到了广泛应用，同时它也是智能设备设计领域最为热门的技术之一。学习和应用嵌入式技术已经成为通信、电子、计算机和自动化领域工程师感兴趣的话题，衷心地希望本书能对相关专业的教师和学生、从事嵌入式系统设计和开发的研究人员及企业工程师有所帮助。

本书由郑州科技学院电气工程学院李杰、刘林阴、陈慧丽共同编写完成。感谢北京理工大学出版社的大力支持，它使本书得以与读者见面。

由于知识所限，书中不足之处在所难免，恳请各位专家和读者指正。

编　者

目录 Contents

第一章 嵌入式操作系统

嵌入式操作系统（Embedded Operating System，EOS）是一种用途广泛的系统软件，过去它主要应用于工业控制和国防系统领域。EOS 负责嵌入式系统的全部软、硬件资源的分配、调度作业，控制、协调并发活动；它必须体现其所在系统的特征，并能够通过装卸某些模块来达到系统所要求的功能。

目前，已推出一些应用比较成功的 EOS 产品系列。随着 Internet 技术的发展、信息家电的普及应用及 EOS 的微型化和专业化，EOS 开始从单一的弱功能向高专业化的强功能方向发展。嵌入式操作系统在系统实时高效性、硬件的相关依赖性、软件固化以及应用的专用性等方面具有较为突出的特点。嵌入式操作系统是嵌入式系统的重要组成部分。

本章主要介绍嵌入式操作系统的特点，并着重介绍嵌入式系统中常用的实时操作系统以及嵌入式系统的设计方法。

1.1 嵌入式操作系统

1.1.1 嵌入式操作系统的特点

传统的操作系统是计算机的一个大型的系统软件，是计算机的管家和指挥中心。通过它可以实现计算机自身硬件、软件的管理，提高计算机资源的利用率，合理地组织计算机的工作流程，增强计算机的处理能力，提供友好的人机界面，方便用户使用计算机。

嵌入式操作系统（EOS）除了具有上述功能之外，还有其自身的特点：

（1）可装卸性。EOS 的体系结构具有开放性、可伸缩性。

（2）强实时性。EOS 实时性一般较强，可用于各种设备控制当中。

（3）统一的接口。EOS 提供各种设备驱动接口。

（4）操作方便、简单，提供友好的图形界面，易学易用。

（5）提供强大的网络功能，支持 TCP/IP 协议及其他协议，提供 TCP/UDP/IP/PPP 协议支持及统一的 MAC 访问层接口，为各种移动计算设备预留接口。

（6）强稳定性、弱交互性。嵌入式系统一旦开始运行就不需要用户过多地干预，这就要求负责系统管理的 EOS 具有较强的稳定性。嵌入式操作系统的用户接口一般不提供操作命令，它通过系统的调用命令向用户程序提供服务。

（7）固化代码。在嵌入式系统中，嵌入式操作系统和应用软件被固化在嵌入式系统计算机的 ROM 中。辅助存储器在嵌入式系统中很少使用，因此，嵌入式操作系统的文件管理功能应该能够很容易地拆卸，而使用各种内存文件系统。

（8）更好的硬件适应性，也就是良好的移植性。

知识点

嵌入式操作系统伴随着嵌入式系统的发展经历了 4 个比较明显的阶段。

第一阶段是无操作系统的嵌入算法阶段，是以单芯片为核心的可编程控制器形式的系统，同时具有与监测、伺服、指示设备相配合的功能。

第二阶段是以嵌入式 CPU 为基础、以简单操作系统为核心的嵌入式系统。

第三阶段是通用的嵌入式实时操作系统阶段，是以嵌入式操作系统为核心的嵌入式系统。

第四阶段是以基于 Internet 为标志的嵌入式系统，这是一个正在迅速发展的阶段。

1.1.2 嵌入式操作系统的分类

一般情况下，嵌入式操作系统可以分为两类，一类是面向控制、通信等领域的实时操作系统，如 WindRiver 公司的 VxWorks、ISI 的 pSOS、QNX 系统软件公司的 QNX、ATI 的 Nucleus 以及经过实时性改造的 Linux 等；另一类是面向消费电子产品的非实时操作系统，这类产品包括个人数字助理（PDA）、移动电话、机顶盒、电子书、WebPhone 等。

1. 非实时操作系统

早期的嵌入式系统中没有操作系统的概念，程序员编写嵌入式程序通常直接面对裸机及裸设备。在这种情况下，通常把嵌入式程序分成两部分，即前台程序和后台程序。前台程序通过中断来处理事件，其结构一般为无限循环；后台程序则掌管整个嵌入式系统软、硬件资源的分配、管理以及任务的调度，是一个系统管理调度程序。这就是通常所说的前、后台系统。一般情况下，后台程序也叫任务级程序，前台程序也叫事件处理级程序。在程序运行时，后台程序检查每个任务是否具备运行条件，通过一定的调度算法来完成相应的操作。对于实时性要求特别严格的操作通常由中断来完成，仅在中断服务程序中标记事件的发生，不再做任何工作就退出中断，经过后台程序的调度，转由前台程序完成事件的处理，这样就不会造成在中断服务程序中处理费时的事件而影响后续和其他中断。

实际上，前、后台系统的实时性比预计的要差。这是因为前、后台系统认为所有的任务具有相同的优先级别，即平等的，而且任务的执行又是通过 FIFO 队列排队，因而那些对实时性要求高的任务不可能立刻得到处理。另外，由于前台程序是一个无限循环的结构，一旦在这个循环体中正在处理的任务崩溃，其将使整个任务队列中的其他任务得不到被处理的机

会，从而造成整个系统的崩溃。由于这类系统结构简单，几乎不需要 RAM/ROM 的额外开销，因而在简单的嵌入式应用中被广泛使用。

2. 实时操作系统

实时系统是指能在确定的时间内执行其功能并对外部的异步事件作出响应的计算机系统。其操作的正确性不仅依赖于逻辑设计的正确程度，而且与这些操作进行的时间有关。“在确定的时间内”是该定义的核心。也就是说，实时系统是对响应时间有严格要求的。

实时系统对逻辑和时序的要求非常严格，如果逻辑和时序出现偏差将会引起严重后果。实时系统有两种类型：软实时系统和硬实时系统。软实时系统仅要求事件响应是实时的，并不要求限定某一任务必须在多长时间内完成；而硬实时系统不仅要求任务响应要实时，而且要求在规定的时间内完成事件的处理。通常，大多数实时系统是两者的结合。实时应用软件的设计一般比非实时应用软件的设计困难。实时系统的技术关键是如何保证系统的实时性。

实时多任务操作系统是指具有实时性、能支持实时控制系统工作的操作系统。其首要任务是调度一切可利用的资源完成实时控制任务，其次才着眼于提高计算机系统的使用效率，其重要特点是要满足对时间的限制和要求。实时操作系统具有如下功能：任务管理（多任务和基于优先级的任务调度）、任务间的同步和通信（信号量和邮箱等）、存储器优化管理（含 ROM 的管理）、实时时钟服务、中断管理服务。实时操作系统具有如下特点：规模小、中断被屏蔽的时间很短、中断处理时间短、任务切换很快。

实时操作系统可分为可抢占型和不可抢占型两类。对于基于优先级的系统而言，可抢占型实时操作系统是指内核可以抢占正在运行任务的 CPU 使用权并将使用权交给进入就绪态的优先级更高的任务，是内核抢了 CPU 让别的任务运行。不可抢占型实时操作系统使用某种算法并决定让某个任务运行后，就把 CPU 的控制权完全交给该任务，直到它主动将 CPU 控制权还回来。中断由中断服务程序来处理，可以激活一个处于休眠态的任务，使之进入就绪态；而这个进入就绪态的任务还不能运行，一直要等到当前运行的任务主动交出 CPU 的控制权。使用这种实时操作系统比不使用实时操作系统的系统性能好，其实时性取决于最长任务的执行时间。不可抢占型实时操作系统的缺点也恰恰是这一点，如果最长任务的执行时间不能确定，系统的实时性就不能确定。

可抢占型实时操作系统的实时性好，优先级高的任务只要具备了运行的条件，或者说进入了就绪态，就可以立即运行。也就是说，除了优先级最高的任务，其他任务在运行过程中都可能随时被比它优先级高的任务中断，让后者运行。这种方式的任务调度保证了系统的实时性，但是，如果对任务抢占 CPU 控制权处理不好，会产生系统崩溃、死机等严重后果。

重点

（1）嵌入式系统的特点：可装卸性；强实时性；统一的接口；操作方便、简单，提供友好的图形界面，易学易用；提供强大的网络功能；强稳定性、弱交互性；固化代码；更好的硬件适应性，也就是良好的移植性。

（2）嵌入式操作系统的分类：非实时操作系统和实时操作系统（可抢占型和不可抢占型）。

1.2 实时操作系统

1.1 节已经简单介绍了实时操作系统的一些基本概念。本节从其他方面继续介绍实时操作系统的相关知识。

1.2.1 实时操作系统与通用操作系统的比较

通用操作系统是由分时操作系统发展而来的，大部分都支持多用户和多进程，负责管理众多的进程并为它们分配系统资源。分时操作系统的基本设计原则是：尽量缩短系统的平均响应时间并提高系统的吞吐率，在单位时间内为尽可能多的用户请求提供服务。由此可以看出，分时操作系统注重平均表现性能，不注重个体表现性能。通用操作系统中采用的很多策略和技巧都体现出了这种设计原则，如虚存管理机制中所采用的 LRU 等页替换算法。

实时操作系统除了要满足应用的功能需求以外，更重要的是还要满足应用提出的实时性要求。而组成一个应用的众多实时任务对于实时性的要求是各不相同的。此外，实时任务之间可能还会有一些复杂的关联和同步关系，如执行顺序限制、共享资源的互斥访问要求等。这就为系统实时性的保证带来了很大的困难。

因此，实时操作系统所遵循的最重要的设计原则是：采用各种算法和策略，始终保证系统行为的可预测性。可预测性是指在系统运行的任何时刻，在任何情况下，实时操作系统的资源调配策略都能为争夺资源（包括 CPU、内存、网络带宽等）的多个实时任务合理地分配资源，从而使每个实时任务的实时性要求都能得到满足。

与通用操作系统不同，实时操作系统注重的不是系统的平均表现，而是要求每个实时任务在最坏的情况下都要满足其实时性要求。也就是说，实时操作系统注重的是个体表现，更准确地讲是个体在最坏情况下的表现。

由于实时操作系统与通用操作系统的基本设计原则差别很大，因此在很多资源调度策略的选择上以及操作系统实现的方法上两者都具有较大的差异，这些差异主要体现在以下几点：

（1）任务调度策略。

通用操作系统中的任务调度策略一般采用基于优先级的抢占式调度策略。对于优先级相同的进程则采用时间片轮转调度方式，用户进程可以通过系统调用动态地调整自己的优先级，操作系统则可根据情况调整某些进程的优先级。

实时操作系统中的目前使用最广泛的任务调度策略主要可分为两种：一种是静态表驱动方式，另一种是固定优先级抢占式调度方式。

静态表驱动方式是指在系统运行前，工程师根据各任务的实时要求用手工的方式或在辅助工具的帮助下生成一张任务的运行时间表。在运行时，调度器只需根据这张表在指定的时刻启动相应的任务即可。

固定优先级抢占式调度方式则与通用操作系统中采用的基于优先级的调度方式基本类似。但在固定优先级抢占式调度方式中，进程的优先级是固定不变的，并且该优先级是在运行前通过某种优先级分配策略指定的。目前市场上大部分的实时操作系统采用的都是这种调度方式。

（2）内存管理。

为解决虚存给系统带来的不可预测性，实时操作系统一般采用如下两种方式：

① 在原有虚存管理机制的基础上增加页面锁定功能。用户可将关键页面锁定在内存中，从而使该页面不会被 swap 程序交换出内存；

② 采用静态内存划分的方式，为每个实时任务划分固定的内存区域。

目前市场上的实时操作系统一般都采用第一种管理方式。

（3）中断处理。

在通用操作系统中，大部分外部中断都是开启的，中断处理一般由设备驱动程序来完成。由于通用操作系统中的用户进程一般都没有实时性要求，而中断处理程序直接跟硬件设备交互，可能有实时性要求，因此中断处理程序的优先级被设定为高于任何用户进程。

但对于实时操作系统，采用上述中断处理机制是不合适的。首先，外部中断是环境向实时操作系统进行的输入，它的频度是与环境变化的速率相关的，而与实时操作系统无关。如果外部中断产生的频度不可预测，则一个实时任务在运行时被中断处理程序阻塞的时间开销也是不可预测的，从而使任务的实时性得不到保证；如果外部中断产生的频度是可预测的，一旦某外部中断产生的频度超出其预测值（如硬件故障产生虚假中断信号或预测值本身有误）就可能会破坏整个系统的可预测性。其次，实时操作系统中的各用户进程一般都有实时性要求，因此中断处理程序的优先级高于所有用户进程的优先级的分配方式是不合适的。

一种较适合实时操作系统的中断处理方式为：除时钟中断外，屏蔽所有其他中断，中断处理程序变为周期性的轮询操作。另一种可行的方式是：对于采用轮询方式无法满足需求的外部事件，采用中断方式，其他时间仍然采用轮询方式。但此时中断处理程序与其他任务一样拥有优先级，调度器根据优先级对处于就绪态的任务和中断处理程序统一进行处理器调度。

此外，为提高时钟中断响应时间的可预测性，实时操作系统应尽可能少地屏蔽中断。

（4）共享资源的互斥访问。

通用操作系统一般采用信号量机制来解决共享资源的互斥访问问题。对于实时操作系统，如果任务调度采用静态表驱动方式，共享资源的互斥访问问题在生成运行时间表时已经考虑到了，因此在运行时无须再考虑。如果任务调度采用基于优先级的方式，则传统的信号量机制在系统运行时很容易造成优先级倒置的问题，即当一个高优先级任务通过信号量机制访问共享资源时，该信号量已被一低优先级任务占有，而这个低优先级任务在访问共享资源时可能又被其他一些中等优先级的任务抢先，因此造成高优先级任务被许多具有较低优先级的任务阻塞，实时性难以得到保证。因此在实时操作系统中，往往对传统的信号量机制进行了一些扩展，引入了如优先级继承协议、优先级顶置协议等机制，从而较好地解决了优先级倒置的问题。

（5）系统调用以及系统内部操作的时间开销。

进程通过系统调用得到操作系统提供的服务。操作系统通过内部操作（加上、下文切换等）来完成一些内部管理工作。为保证系统的可预测性，实时操作系统中的所有系统调用以及系统内部操作的时间开销都应是有界的，并且该界限是一个具体的量化数值。而在通用操作系统中对这些时间开销则未作如此限制。

（6）系统的可重入性。

在通用操作系统中，核心态系统调用往往是不可重入的。当一低优先级任务调用核心态

系统调用时，在该时间段内到达的高优先级任务必须等到低优先级的系统调用完成才能获得CPU，这就降低了系统的可预测性。因此，实时操作系统中的核心态系统调用往往被设计为可重入的。

知识点

使用实时操作系统的必要性：

首先，嵌入式实时操作系统提高了系统的可靠性。

其次，嵌入式实时操作系统提高了开发效率，缩短了开发周期。

再次，嵌入式实时操作系统充分发挥了 32 位 CPU 的多任务潜力。

1.2.2 实时操作系统的评价指标

由于实时操作系统在实时应用中的特殊地位，对其性能指标的要求相对于通用操作系统来说更加严格。

评价一个实时操作系统，一般可以从进程管理、任务调度、内存管理、任务通信、内存开销、任务切换时间和最大中断禁止时间等技术指标来衡量其优劣。

下面具体介绍这些性能指标：

（1）任务调度算法。

实时操作系统的实时性和多任务能力在很大程度上取决于它的任务调度算法。任务调度算法可以有以下分类方式：从调度策略来讲，可分为优先级调度策略和时间片轮转调度策略；从调度方式上来讲，可分为可抢占式、不可抢占式以及选择克抢占式三种调度方式；从时间片来看，可分为固定与可变时间片轮转两种方式。

（2）内存管理和最小内存开销。

在内存管理方面，可分为实模式与保护模式两种。

在实时操作系统的设计过程中，最小内存开销是一个较重要的指标。这是因为在工业控制领域中，出于降低成本的考虑，某些工控机的内存配置一般不大。而在有限的空间内不仅要装载实时操作系统，还要装载用户程序。这是实时操作系统与通用操作系统设计的明显区别之一。

（3）确定性。

在实时操作系统中，在一定的条件下，系统调用运行的时间应该是可以预测的，但这并不意味着所有的系统调用都总是执行一个固定长度的时间，而不管系统的负载如何。但是，系统调用的最大执行时间可以确定。

（4）任务切换时间。

当由于某种原因一个任务退出运行时，实时操作系统要保存它的运行现场信息，插入相应队列，并依据一定的调度算法重新选择一个任务使之投入运行。这一过程所需时间称为任务切换时间。

（5）最大中断禁止时间。

当实时操作系统运行在内核态或执行某些系统调用时，是不会因为外部中断的到来而中断执行的。只有当实时操作系统重新回到用户态时才会响应外部中断请求。这一过程所需的最大时间就是最大中断禁止时间。

上述几项指标中，最大中断禁止时间和任务切换时间是评价实时操作系统性能最重要的两个技术指标。

重点

实时操作系统与通用操作系统的差异表现在：任务调度策略；内存管理；中断处理；共享资源的互斥访问；系统调用以及系统内部操作的时间开销；系统的可重入性。

实时操作系统的评价指标：任务调度算法；内存管理和最小内存开销；确定性；任务切换时间；最大中断禁止时间。其中，最大中断禁止时间和任务切换时间是评价实时操作系统性能最重要的两个技术指标。

1.3　基于 Linux 的嵌入式操作系统

Linux 是一套由芬兰赫尔辛基大学（University of Helsinki）的学生 Linus Torvalds 开发的系统内核（kernel）演化而成的电脑操作系统。1991 年，Linus（当时 21 岁）为了学习需要，在 80 386 平台上开发了类似 UNIX 和 MINIX（教育用的 UNIX 兼容操作系统）的操作系统，并在互联网上发布源代码。此后，其吸引了大量的开发志愿者和黑客加入了 Linux 内核的开发了行列。1994 年，1.0 版本的 Linux 内核正式完成。如今，经过十多年的努力，Linux 已被应用到多个领域，小至手机、PDA 等嵌入式系统，大至拥有上千个主机的超级电脑及银行、太空实验等要求极高稳定性的高端系统。

除内核之外，Linux 系统的其他部分主要是由 Richard Stallman 及其领导的自由软件基金会（Free Software Foundation，FSF），以及无数开发志愿者和黑客共同研发的成果。FSF 在 1984 年建立了 GNU 项目，目的是开发一个兼容 UNIX 的操作系统。而 Linux 的开发也是基于 FSF 所提供的自由软件，并且加入了大量 GNU 的软件开发成果，故也被称为 GNU/Linux。

Linux 是一个成熟而稳定的网络操作系统。将 Linux 植入嵌入式设备具有许多优点。首先，Linux 的源代码是开放的，任何人都可以获取并修改，用之开发自己的产品。其次，Linux 是可以定制的，其系统内核最小只有约 134 KB。一个带有中文系统和图形用户界面的核心程序也可以做到不足 1 MB，并且同样稳定。另外，它和多数 UNIX 系统兼容，应用程序的开发和移植相当容易。同时，由于 Linux 具有良好的可移植性，人们已成功使 Linux 运行于数百种硬件平台之上。

然而，Linux 并非专门为实时性应用而设计，因此如果想在对实时性要求较高的嵌入式系统中运行 Linux，就必须为之添加实时软件模块。这些模块运行的内核空间正是操作系统实现进程调度、中断处理和程序执行的部分，因此错误的代码可能会破坏操作系统，进而影响整个系统的可靠性和稳定性。

尽管如此，Linux 的众多优点使它在嵌入式领域获得了广泛的应用，并出现了数量可观的嵌入式 Linux 系统。其中具有代表性的包括：ARMLinux、uClinux、ETLinux、ThinLinux、LOAF 等。

下面着重介绍应用最为广泛的 ARMLinux 和 uClinux。

1.3.1 ARMLinux 简介

完整的嵌入式 Linux 解决方案应包括嵌入式 Linux 操作系统内核、运行环境、图形化界面和应用软件等。由于嵌入式设备的特殊要求，嵌入式 Linux 解决方案中的内核、环境、GUI 等都与标准 Linux 有很大不同，其主要挑战是如何在狭小的 FLASH、ROM 和内存中实现高质量的任务实时调度、图形化显示、网络通信等功能。

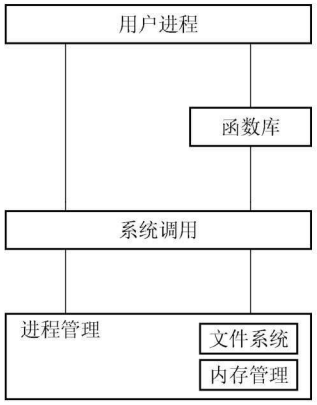

图 1.1　Linux 内核结构体系

Linux 内核有自己的结构体系，其中进程管理、内存管理和文件系统是其最基本的 3 个子系统。图 1.1 简单地表示了它的框架。用户进程可直接通过系统调用或者函数库来访问内核资源。正因为 Linux 内核具有这样的结构，因此修改内核时必须注意各个子系统之间的协调。

1. 裁剪内核

嵌入式 Linux 内核一般由标准 Linux 内核裁剪而来。用户可根据需求配置系统，剔除不需的服务功能、文件系统和设备驱动。经过裁剪、压缩后的系统内核一般只有 300 KB 左右，十分适合嵌入式设备。

对标准 Linux 的修改主要是虚拟内存和调度程序部分的改动。对于实时性要求较高的嵌入式系统来说，实时任务往往要求 CPU 具有很高的突发处理能力，即在有些时候需要极高的处理效率，因此需要屏蔽内核的虚拟内存管理机制。

强实时需求的嵌入式应用可以通过修改任务调度模块实现，主要是在内核和设备驱动程序中加入许多切换点。在该点处，系统检测是否存在未处理的紧急中断，有则剥夺内核的运行，及时处理中断。实现实时性服务的一个较好的方法是在标准的 Linux 内核上增加一个实时内核，标准 Linux 内核作为一个任务运行于实时内核上，强实时性任务也直接运行在实时内核上，如 RT-Linux 等。

文件系统是嵌入式 Linux 操作系统必不可少的。一般嵌入式设备文件系统主要使用 RamDisk 技术和网络文件系统技术。RamDisk 可驻留于 FLASH，运行时加载到内存中。

2. 精简运行环境

Linux 通常的运行环境指用户运行任何应用的基础设施，主要包括函数库和基本命令集等。

Linux 应用运行所需的函数库主要有 C 库、数学库、线程库、加密库、网络通信库等。其中最基本的是 C 语言的运行库 glibc。这个库主要完成基本的输入/输出、内存访问、文件处理等功能。一个标准的 glibc 库大约需要 1 200 KB 存储空间，考虑到嵌入式 Linux 内核往往很小，这种运行库实在太大，因此需要做一些精简的工作，方法有两种：① 使用静态链接的方法，完全不使用运行库动态链接；② 对这个库的函数进行精简。在嵌入式系统中，很少有多个程序并行的可能，程序的维护，尤其是库函数的维护更新是不常见的。这时，使用静态链接的优势就极为明显。为了便于将来扩充的需要，采用第二种方法，针对需要，对库函数的内容进行精简，只保留一些基本功能。

基本命令集同样是运行用户应用的基础，主要包括初始化进程 init、终端获取 getty、shell 和基本命令等。嵌入式系统的启动过程可能与标准 Linux 不同，例如跳过登录过程直接启动