
# 嵌入式操作系统基础 μC/OS-Ⅱ和Linux（第2版）

任 哲 樊生文 编著

北京航空航天大学出版社
BEIHANG UNIVERSITY PRESS

# 嵌入式操作系统基础
# μC/OS－II 和 Linux
# （第2版）

任　哲　樊牛文　编著

北京航空航天大学出版社

## 内 容 简 介

本书为《嵌入式操作系统基础 μC/OS-II 和 Linux》的第 2 版，本书继承了第 1 版的写作风格，仍以操作系统的初学者和高等院校的学生为对象，首先介绍便于学习和理解的微内核 μC/OS-II，从而使读者快速了解嵌入式实时操作系统的主要组成、功能及特点；然后从主要数据结构的角度介绍源码开放的操作系统 Linux；最后，简单地介绍 Linux 的嵌入式版本 μCLinux。与第 1 版相比，本书在重点改写 Linux 大部分内容的同时增加了必要的例题，从而大大地降低了初学者的学习难度。

本书可作为高等院校课程的教材或教学参考书，同时也适合对嵌入式操作系统感兴趣的工程技术人员阅读、参考。

**图书在版编目(CIP)数据**

嵌入式操作系统基础 μC/OS-II 和 Linux / 任哲，樊生文编著. --2 版. -- 北京 ：北京航空航天大学出版社，2011.8

ISBN 978-7-5124-0549-3

Ⅰ.①嵌… Ⅱ. ①任… ②樊… Ⅲ. ①实时操作系统－程序设计②Linux 操作系统 Ⅳ. ①TP316.2②TP316.89

中国版本图书馆 CIP 数据核字(2011)第 153005 号



**嵌入式操作系统基础 μC/OS-II 和 Linux(第 2 版)**

任 哲 樊生文 编著

责任编辑 王慕冰 龚荣桂 朱胜军

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) http://www.buaapress.com.cn

发行部电话：(010)82317024 传真：(010)82328026

读者信箱：bhpress@263.net 邮购电话：(010)82316936

北京时代华都印刷有限公司印装 各地书店经销

*

开本：787×960 1/16 印张：34.0 字数：762 千字

2011 年 8 月第 2 版 2011 年 8 月第 1 次印刷 印数：4 000 册

ISBN 978-7-5124-0549-3 定价：62.00 元

# 第 2 版前言

在《嵌入式操作系统基础 μC/OS－II 和 Linux(第 2 版)》出版之际，想说如下几句话：

一、很高兴《嵌入式操作系统基础 μC/OS－II 和 Linux》对读者有帮助，也很高兴出版该书的第 2 版。本书主要在 Linux 的介绍方面进行了较大幅度的修改。

二、感谢读者多年来为第 1 版提出的意见和建议，这些意见和建议或多或少地指导了第 2 版的修改。

三、再次提醒各位读者，这本书仅是一本入门书，其宗旨是引导读者比较顺利地使用其他文献学习操作系统，因此不可能面面俱到。本书仅在作者认为是学习难点的知识点上进行了相关的讲解。

四、关于操作系统以及 Linux 的学习问题，在本人编写的另一本教材《微型计算机操作系统基础——基于 Linux/i386》的前言中，曾以回答问题的方式做了如下表达：

**1. 对于搞顶层软件设计的人来说，学习操作系统有意义吗？**

太有意义了，因为你设计的软件在运行时，操作系统是你的应用程序的一部分(只不过不是你设计的罢了)。如果你对使用的操作系统不了解，那你知道它为你做了些什么你不需要的工作？你用什么向客户说明你的软件是安全的？你用什么来保证当系统升级时，你的软件是兼容的？

**2. 对于搞硬件的人，学习操作系统有意义吗？**

现在搞硬件的人是没有办法不接触软件的。试想，要设计一个硬件驱动程序，你不了解操作系统的设备管理能行吗？尤其是在嵌入式系统应用如此广泛的今天，你具备嵌入式操作系统的选用、裁剪及改造能力吗？你能成为一个合格的嵌入式工程师吗？

**3. 现在为什么都在介绍 Linux，而不介绍 Windows 及其他操作系统呢？**

没办法，因为教学上作为依托的软件实例必须是源码开放的，否则会涉及知识产权问题。就目前情况来看，适合作为计算机专业教学使用且源码开放的只有 Linux，对其他操作系统则只能就其公开的技术进行一般的介绍。

**4. Linux 具有普遍意义吗？学了它就能把其他系统也搞明白吗？**

这个问题比较复杂。确实，作为教学使用的实例最好具有普遍意义，以达到解剖一个实例就可以解决该课程全部问题的目的。就这一点来看，Linux 并不理想，至少在操作系统中比较重要的微内核概念就不能通过它来体现。另外，Linux 的进程、线程的概念不很清晰(但很实

用),在学生学习阶段容易造成一些不必要的混乱。但由于操作系统是一个实用系统,只要是具体系统就免不了"各有各的高招",对于计算机这种非线性系统来说,不同招数之间的差异还很大。因此,企图以一个实例就把所有操作系统搞清楚的想法是不实际的,加之前面所说的源码开放问题,现在的教学实例也只能是 Linux 了。但由于操作系统的基本作用和原理都大致相同,所以学习了 Linux 之后,学习其他系统也就不难了。

**5. 学习了 Linux 之后那些比较注重理论的书还要看吗?**

理论总是重要的,它是实际系统的抽象和凝炼,具有普遍的指导意义。如果读者真想研究操作系统,那么那些对你来说目前似乎没有什么用的理论,将来是会有大用途的。特别是一些人们在长期实践中总结出的设计模式(可以视为理论吧)很有用。

**6. 学习操作系统能提高编程能力吗?**

能。前面已经讲过了,操作系统就是一个真实的大程序实例。在相关课程中很难用实例介绍的数据结构、算法、设计模式,以及软件工程中提到的"高内聚、低耦合"软件设计原则、代码动态链接以及充分利用指针实现的各种虚拟技术等,在操作系统中都有极为充分的体现,如果注意总结,那么你的软件设计能力会有突飞猛进的提高。

**7. 学习 Linux 难不难?**

入门,甚至应用都不难,但要精通确实很难。想想看,从 Linux 出现到现在,从程序设计方法上,它历经了面向过程程序设计、面向对象程序设计两大阶段;从内核结构上,尽管它很难改变当初的宏内核设计,但后来的一些内核新概念,例如层次化内核和微内核,也给了它极大的影响,加之为了嵌入式应用,它还在实现实时内核方面做出了很大努力。因此可以说,Linux 是集各种编程思想和方法之大成者。因为 Linux 所采用技术之广泛和繁杂,要精通它确实很难,所以从这个角度看,它并不是一个特别理想的教学实例。在此作者也建议初学者在阅读本书的同时,最好看一些介绍其他小型操作系统的书籍和资料,例如开源的 μC/OS－II、T－KERNEL 等,以快速了解操作系统的一些基本概念。另外,在阅读本书介绍的一些大型数据结构时,也需要某种程度的"不求甚解",以把握其中的主要内容。

**8. 这本书适合哪类读者?如何学习 Linux?**

学无定法,作者只能提一些建议。如果只是单纯以一般的顶层应用为目的,那么操作系统就是一堆函数,你把这些函数的用途及使用方法弄清楚也就可以了,这本书可能不适合你。但要想对操作系统的用途、原理、特点和实现方法有一定程度的了解或是想入门,这本书就比较合适,因为这本书尽量做到注重说理、深度适中、文字通俗易懂。当然,要深入研究,单靠这一本书就远远不够了,还需要看一些理论性较强或对代码剖析得比较详细的书籍。

至于学习方法，则建议如下：如果有条件，最好是以 Shell 编程来熟悉 Linux，以阅读数据结构来了解内核架构，以编写守护进程和驱动程序为突破点来理解 Linux 内核。

五、由于作者的水平及文字能力所限，本书大量地引用了其他作者及网上的一些资料，在此向这些作者致以衷心的感谢。

六、第 2 版仍然还会出现各种各样的错误和问题，敬请读者来信指正。

本书的作者为樊生文、房红征、潘树林和任哲，其中樊生文与任哲为主编并负责统稿。樊生文编写了第 14～22 章，任哲编写了第 1～10 章，房红征编写了第 12 和第 13 章，潘树林编写了第 11 章。

作者的电子邮箱为 renzhe71@sina.com。

作　者
2011 年 5 月 11 日

# 前　言

众所周知，把计算机系统嵌入到某个宿主对象（例如，多媒体设备、洗衣机、照相机、空调器、飞机、汽车等）中，会使对象的自动化程度大为提高，并使之在功能和性能两方面发生革命性的飞跃，这无疑是对人们的一个不可抗拒的诱惑。

因此，无论是生产厂商还是系统开发人员，都对嵌入式系统的研究、开发和学习投入了极大的热情。当然，其中也包括了高等院校的学生和教师。很自然地，就引发了关于什么是嵌入式系统，嵌入式系统具有什么样的特点以及如何在相关专业的教学体系中引入嵌入式系统教学内容的讨论。

在讨论中，作者支持这样一种观点：嵌入式系统仍然是计算机系统，只不过它是具有应用上的一些特点，以及为适应嵌入式应用而在处理器结构及其软件上与通用计算机相比较有某种变化的计算机系统罢了，没有必要大惊小怪。当然，面对嵌入式系统的广泛应用和我国嵌入式系统开发人员极度匮乏的现状，作为以培养人才为目的的高等院校必须认真对待，并需要立即对现有的计算机教学体系进行必要的改变，以适应当前形势的需要。

基于上面的想法，在对高等院校相关专业的课程体系及课程内容进行了仔细的分析，并与同行进行了广泛的讨论之后，作者认为在以下三方面对现有课程体系及课程内容进行适当的改变，就足以满足嵌入式系统教学的需要：

- 改造计算机硬件课程内容。目前，嵌入式处理器在硬件功能上已经比较接近通用处理器芯片了。因此，对于计算机专业来说，在现有的计算机硬件课程的基础上，增加嵌入式处理器的内容是完全可行的；对于自动化、仪器仪表、电子技术等非计算机专业来说，则可以把原来的单片机课程改为以嵌入式处理器（系统）为主要内容的课程，而把现在应用面仍然很广且适合低端嵌入式应用的单片机（单片机也是嵌入式应用）的内容进行适当压缩，作为辅助内容让学生自学。
- 根据嵌入式系统的开发特点，增设一门以介绍实时系统、交叉编译及代码优化等与嵌入式系统软件开发技术方面为主的课程（最好通过设计类教学环节来完成）。
- 大幅度改革计算机操作系统课程。作者之所以主张要大幅度改革，其理由有二：一是原来这门课程的问题就较大，其主要问题就是课程内容太抽象、太空洞，也太繁杂，用学生的话来说是学了跟没学差不多；二是基本没有什么实践教学环节。而对于大多数非计算机专业来说，原来根本就没有操作系统这门课程，但是，如果要搞嵌入式系统，则不把操作系统的知识掌握到一定程度是绝对不行的。

基于上述想法，为了使人们对操作系统有一个较为系统的了解和认识，作者曾选择了在我国已经具有一定影响且很实用的 μC/OS－II 嵌入式实时操作系统内核，根据邵贝贝老师翻译的《嵌入式实时操作系统 μC/OS－II》(参见参考文献[2])，按照自己对教材的理解编写了一本自认为适合高等院校非计算机专业学生使用的教材《嵌入式实时操作系统 μC/OS－II 的原理及应用》(参见参考文献[13])。

没有想到，这本书出版不久，作者就陆续接到了一些读者的来信，希望作者再编写一部比较通俗的有关嵌入式 Linux 的书籍或者教材。

但是，凡是从事过操作系统课程教学和编写过相关教材的人都知道这样一个事实：上述这两项工作，无论干哪一项都不会有什么好结果。原因很简单：操作系统的内容既涉及硬件又涉及软件，使用的技术又几乎覆盖了计算机技术的所有方面。这样，就使得在现有的教学学时及教材的有限篇幅里，把这门课程的内容讲清楚，并被学生和读者所掌握就成为了一件极为困难事情。本人作为一名教师，当然知道其中的难处。但是，鉴于 Linux 这种通用操作系统在嵌入式应用中的良好表现，计算机专业的学生学习 Linux 很有必要，再加上本人教学上的需要，思虑再三，就试着编写了本书。

由于 Linux 实在太大了，用它来建立操作系统的概念确实很困难。如果不分析代码，则又是空洞无比；如果分析代码，则又会被代码的汪洋大海所淹没而难有出头之日。因此，本书还是以精干小巧的微内核 μC/OS－II 为切入点，使读者能快速建立起操作系统的调度、通信、同步等一些基本概念。有了这些概念之后，再来对付 Linux 这个庞然大物。而对于 Linux，则以入门为主要目标，以介绍 Linux 用来描述操作系统各部分的主要数据结构为主，力求使读者对 Linux 的组成与结构有一个较清晰的了解和把握，从而为读者进一步研究 Linux 打下基础。另外，作为一本操作系统内核入门的书籍，本书在文字和语言上尽可能做到通俗易懂。

全书内容共 22 章，参加本书编写的作者有潘树林、房红征和任哲，由任哲担任主编并负责全书的统稿工作。

在本书的策划和编写过程中，得到了很多同志的关心、指导和帮助。特别是在本书的第 13 章关于μC/OS－II 在 ARM 上移植中使用了周立功先生的方法和程序代码，在部分章节还使用了 Jean J. Labrosse 先生书中的部分例题代码。同时，在编写本书的过程中，作者还参阅了大量的参考书籍，并在本书中引用了这些书籍的一些文字和插图。为此，本书作者向为本书作出贡献的人们表示衷心的感谢！

由于作者在嵌入式系统知识方面的欠缺，尤其是在嵌入式系统教学方面的不足，因此本书无论是在教学内容的选取编排上，还是对于难点、重点的讲解上，一定会有很多不足甚至是错误，在此诚挚希望读者能提出批评指正。作者的电子邮箱为 renzhe71@sina.com。

任　哲

2006 年 4 月 3 日

# 目　录

# 第1章　操作系统的基本概念

操作系统(Operating System,OS)是计算机的一种重要系统软件。它屏蔽了计算机硬件的一些细节,并通过应用程序接口(Application Programming Interface, API)向用户提供通用服务,从而使应用程序设计人员得以在一个友好的平台上进行应用程序的设计和开发,大大提高了开发效率。

本章的主要内容有:

- 计算机操作系统的基本概念;
- 应用程序接口、系统调用及其实现;
- 并发执行及进程和线程的基本概念;
- 进程调度和进程切换;
- 进程的同步和通信;
- 存储管理、I/O 设备和文件管理的基本知识;
- 宏内核和微内核的基本概念;
- 操作系统的引导、装载和启动;
- 操作系统的分类。

## 1.1　计算机操作系统

操作系统是计算机系统硬件的软件封装,它使计算机成为功能强大的虚拟计算机,从而使应用程序可以使用软件的方式使用计算机的底层功能。为提高应用程序的执行效率和增强计算机的安全性能,它还负责替用户对计算机系统的资源进行管理。

### 1.1.1　什么是计算机操作系统

众所周知,计算机是一种功能强大的数字运算装置。作为一种装置,它需要用诸如中央微处理器(CPU)、存储器、接口及外部设备等一些实际物理装置来构成。这些实际的物理装置就构成了计算机的硬件系统。

单纯由硬件构成的计算机叫做“裸机”。裸机只提供了计算机系统的物质基础,单靠硬件

是不能完成用户所要求的数字运算工作的，只有在计算机硬件的基础上再配以相应的软件，它才是一台真正的计算机系统，才能完成人们所交付的各种运算任务。

现在，人们经常要在计算机上使用 Microsoft Word 之类的软件来书写和编辑文档，也经常使用 Microsoft Excel 之类的软件做一些数据的管理和统计工作。当人们在计算机上安装 Microsoft Word 或 Microsoft Excel 这些软件时就会被告知，要先看一看计算机上是否装有 Windows。为什么呢？因为如果没有安装 Windows，那么无论是 Microsoft Word 还是 Microsoft Excel 都是不能运行的，因为它们都是以 Windows 为平台来运行的，或者说 Microsoft Word 或 Microsoft Excel 这些软件是必须通过另外一个软件——Windows 才能使计算机硬件工作的。这里的 Windows 就是由微软公司开发的一种计算机操作系统，而 Microsoft Word 和 Microsoft Excel 则是在 Windows 平台上的应用程序。所以，计算机只有安装了某种操作系统之后，才能称做是一个完整的、真正的计算机系统。

计算机的硬件、操作系统与应用程序之间关系如图 1－1 所示。

大体上来说，一个完整的计算机系统是由硬件和软件两大部分组成的。硬件是软件运行的物质基础，软件能充分发挥硬件的潜能和扩充硬件的功能并能完成用户所交付的任务，两者互相依存，缺一不可。

计算机的软件部分又分为应用软件和系统软件两种。

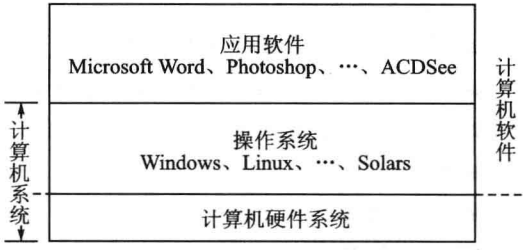


**图 1－1　计算机硬件、操作系统和应用软件之间的关系**

应用程序是属于应用软件范畴的，它们是用来实现某种特定任务的软件，例如前面提到的用于处理文档的 Microsoft Word 和用于处理数据的 Microsoft Excel 等。

不以某种特定应用为目标的软件是系统软件。系统软件是在计算机硬件基础上为应用软件提供通用服务的软件。它们作为一个必需的组成部分，与硬件一起构成了完整的计算机系统，所以它们被叫做系统软件。

操作系统就是诸多系统软件中最重要的一种系统软件，它是处在计算机硬件与计算机应用程序之间，除了起着应用程序与计算机硬件联系的接口作用之外，还要负责对计算机的资源在应用程序之间进行管理和分配。

## 1.1.2　操作系统的功能

可以从两方面来认识操作系统的功能：一是对计算机硬件的封装和功能的扩充；二是它是计算机各种资源的管理者和分配者。

### 1. 操作系统是计算机硬件的封装和功能的扩充

通常，计算机用户是用高级语言来编写应用程序的，但计算机的硬件却是按照机器码指令

来执行操作的。于是，就出现了这样一个问题：用户如何才能用高级语言来指示一台只能识别机器码指令的机器来工作。显然，在高级语言程序和机器硬件之间，必须有一个中介来完成高级语言与低级语言的对接工作。

为了说明问题，先看一下当一个用户面对一台裸机时他会面对什么样的问题及困难。在使用计算机时，如果用户面对的是一台只由硬件组成的裸机，那么用户就不得不使用低级语言来编写指挥硬件的程序。例如，需要从磁盘中读取一批数据，那么凡是涉及读取磁盘数据工作的每一个步骤和细节，包括给出磁头号、启动驱动步进电机并命令磁头移动到给定的磁道位置、给出扇区号、等待磁头和扇区移动到合适位置、读出数据等一系列的繁杂动作，就都需要由用户自己来编写程序。诸如此类的事情，在计算机应用中还有很多。例如，键盘数据的读取、鼠标状态的读取、向打印机输出数据、向磁盘输出数据、向显示器输出数据、计算机硬件异常的处理，等等。显然，上述这些程序的设计工作对于普通计算机用户来说，是极其困难和艰巨的。因为他必须既通晓计算机硬件的所有技术细节，又要精于汇编语言程序设计。

但是，人们发现，这些实现硬件操作的汇编语言程序模块都有一个共同的特点，即它们都具有很强的通用性，也就是说，它们是大多数应用程序都要用到的通用功能。于是，请一些通晓计算机硬件工作机理并精于汇编语言程序设计的人来编写这些程序功能模块，通过这些模块与高级语言对接的接口向用户提供服务，并把这些模块作为一种通用软件提供给用户是一个好办法。这样，用户在装有这种通用软件的计算机上来编写高级语言程序也就非常容易和方便了。

例如，还是上面所说的从磁盘中读取一批数据这项工作，如果系统中已经有了三个具有接口的汇编语言程序模块：磁头移动并定位模块、读磁盘数据模块和写入磁盘模块，那么用户的工作就简单多了，他只要在自己的应用程序中，通过接口调用所需要的模块即可。

可以看到，如果系统中对所有硬件进行操作的程序功能模块都具备，那么应用程序设计人员所面对的就再也不是那些陌生的硬件电路，而是一些见到后就会备感亲切的软件接口了。

这种提供了一些例程接口，从而使应用程序可通过这些接口对计算机硬件进行操作的软件，就叫做计算机硬件的抽象层 HAL。这个层次作为操作系统的最底层，是对计算机硬件的第一次软件封装。

更进一步，如果在硬件抽象层与应用程序之间再添一层；或者说，在硬件抽象层的基础上再加一层，即使用高级语言再编写一些程序模块（例如 C 函数），并在这些模块中，通过调用诸如磁头移动、定位、读磁盘数据和写入磁盘等这些底层程序模块，把这些繁杂的硬件操作根据需要适当地组合封装起来，形成一些有通常意义的模块，那么用户就更加方便了。

例如，可在函数 read()中调用两个汇编例程：磁头移动并定位和读磁盘数据。那么，用户应用程序在需要读取磁盘数据时，就可以简单地提出类似如下请求：

```
read(fp,buf,count);
```