HZ BOOKS

重点大学
计算机教材

Windows
内核实验教程

陈向群 马洪兵 王 雷
林 斌 王 磊 张 高 编著

《Windows操作系统原理》配套实验教程

机械工业出版社
China Machine Press

重点大学计算机教材

Windows内核实验教程

陈向群 马洪兵 王 雷
林 斌 王 磊 张 高 编著

机械工业出版社
China Machine Press

本书是在微软亚洲研究院和美国微软公司的支持下，由美国微软公司全面提供Windows内部技术资料，全国知名重点大学操作系统主讲教师组成写作组编撰的以Windows 2000/XP为实际示例，讲授操作系统原理实验课程的教科书。

本书基于Windows 2000/XP设计了一组操作系统课程实验，这些实验与操作系统课程的教学内容相对应。实验的安排循序渐进，很好地适应了课程的学习曲线，并对实验涉及的相关原理性内容进行了铺垫。本书突出的特点是：实验内容与课程教学相呼应，经典内容与现代发展并举，掌握基本方法与提高技术水平并重，原理与源代码相结合。

本书适合作为高等院校计算机科学技术和电子信息类专业操作系统实验课程的教材，也是设计、开发基于Windows 2000/XP平台应用程序及操作系统驱动程序的重要参考书。

图书在版编目（CIP）数据

Windows内核实验教程/陈向群等编著. -北京：机械工业出版社，2002.9
（重点大学计算机教材）
ISBN 7-111-10880-9

Ⅰ. W… Ⅱ. 陈… Ⅲ. 操作系统原理，Windows－实验－高等学校－教材 Ⅳ. TP316. 7

中国版本图书馆CIP数据核字（2002）第064898号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）
责任编辑：杨海玲
[illegible] 印刷·新华书店北京发行所发行
2002年9月第1版第1次印刷
787mm × 1092mm 1/16 · 14.75印张
印数：0 001- 5000册
定价：25.00元（附光盘）

序

史美林

在当今的知识经济时代中，计算机技术的发展和普及对我国信息化进程具有举足轻重的作用，直接影响到信息化进程和国民经济的可持续发展。软件技术是计算机技术中关键技术之一，软件存在于所有的计算机应用系统中，是计算机系统的中枢神经，控制着整个计算机系统。

作为计算机软件的核心组成部分，操作系统是所有计算机系统的基础和平台，它管理着计算机系统中的各种软硬件资源，向应用软件提供基本的开发和运行环境。先进的操作系统技术可极大地改善应用软件的支撑环境，提高整个系统的效率。应用软件的开发也只有充分利用操作系统提供的开发和运行环境，才能有效地实现自己的功能。计算机系统的最终用户也需要了解操作系统的基本概念和操作方法，以便有效地利用计算机来完成自己的工作。可以这样说，不管是操作系统的开发人员、应用软件开发人员，还是计算机的最终用户，都需要或多或少地掌握操作系统的知识。

操作系统课程一直是信息技术专业，特别是计算机专业的基础课。操作系统课程是一门综合性和实践性较强的课程，任何一个实际的操作系统都是十分复杂的系统。如何让学生在较短的时间内很好地掌握操作系统的精髓，并能把这些知识应用于软件应用和开发中一直是操作系统课程教学探索的难点。许多操作系统课老师都一直致力于设计一些好的实验环节，帮助学生理解和掌握复杂的操作系统的工作原理。

本书的作者都是在高等学校里从事操作系统课程教学和相关研究工作的教师。为了使这本操作系统实习教材比较符合我国多数高等学校电子信息类专业的实际情况，他们在调查研究的基础上，拟订了写作原则和内容大纲，并进行了反复的修改，以便能更好地符合实际教学实习的需要。

本书基于Windows 2000/XP设计了一组操作系统课程实验，这些实验与操作系统课程的教学内容相对应。实验的安排是循序渐进的，既有针对一堂课内容的小实验，也有训练综合能力的大实验，并对实验所涉及的相关内容进行了

前 言

操作系统是一门实践性很强的学科。如果只学习操作系统原理而不进行操作系统的具体设计实践，是很难真正掌握操作系统这门学科的精髓的。在学习和掌握操作系统原理的过程中，我们强调进行操作系统设计实践的重要性，是操作系统本身特点的要求。

我们都知道，在当代计算机系统中，操作系统是计算机软件的核心，是所有计算机系统的基础和支撑。在当今社会中，使用一台没有配置操作系统的裸计算机是难以想像的。操作系统管理和控制计算机系统中的所有软、硬件资源，是计算机系统的灵魂和核心。除此之外，操作系统还为用户使用计算机提供一个方便、灵活、安全、可靠的工作环境。因此，学习并掌握计算机操作系统的基本原理，不仅对电子信息类专业的学生和研究人员是必要的，而且对一般计算机应用人员也是非常有益的。

随着计算机应用的普及深入，Microsoft Windows和Linux 等操作系统的名称已经成为日常生活中的常用词语，人们对操作系统学习的愿望也更为强烈。在高等学校中，开设操作系统课程的专业已经不单单局限于电子信息类专业。不仅仅大学本科的学生学习操作系统课程，许多大专甚至中专学校的学生都在学习操作系统课程，而且还有更多的人在自学有关操作系统的原理和知识。操作系统借着计算机应用的普及热浪，进入了某种程度的普及阶段。

操作系统是计算机系统的基础、支撑和核心。但是，处于计算机系统基础、支撑和核心的操作系统却又具有概念抽象、结构复杂和难于掌握的特点。

在我们接触到的国内重点大学本科计算机软件专业的学生中，有不少学生害怕操作系统课程，部分学生不敢选修操作系统实习课程，有的学生甚至在选修了操作系统实习课程之后，再硬着头皮去找老师退修这门课程。还有的学生发出了“操作系统像哲学”的感慨，其含义就是操作系统的概念和原理太让人琢磨不透了。

从当代计算机离不开操作系统的角度来看，操作系统确实有一点类似于哲学。哲学的原理是普遍适用、到处存在的。在当代社会里，只要有人群的地方就有计算机，甚至没有人群的地方也可能有计算机。有计算机的地方，就有操

作系统。从这个角度看，操作系统原理也是普遍适用、到处存在的。哲学是指导人们的社会实践的。在计算机领域中，操作系统理论也是指导实践的。操作系统必须通过实践来学习。从实践的角度来看，操作系统也是某种“实践的自然哲学”。

从操作系统的发展历史来看，早期的计算机系统并没有操作系统。人们在使用计算机系统的过程中，为了提高工作效率，更好地利用机器资源，逐渐设计和发展了操作系统的早期原型，并且在实践的基础上，抽象出有关操作系统的基本原理、概念，并且最终形成了操作系统这样一门理论和实践结合得非常紧密的学科。

如果读者有幸到国际一流大学去学习操作系统课程，就会发现，在所有这些国际一流大学的操作系统课程中，学生都毫无例外地必须参加设计小型操作系统的实习工作。操作系统实习的结果，在学生操作系统课程的成绩中权重极大。

为什么国际一流大学的操作系统课程如此重视操作系统实习？答案很简单，因为学习掌握操作系统的最好途径就是实践。只要进行实践，读者就会发现操作系统的概念并不是那么抽象的，它的结构也是可以逐步分析和理解的。操作系统中的不少概念和原理实际上与人们日常生活中的体验是相通的。这是在几十年操作系统的发展中已经反复证明了的。虽然一门课程的分数并不能完全说明问题，但是一个学生能否出色地完成操作系统实习，确实反映了这个学生对基础理论的掌握程度和程序设计水平的高低。

由于各方面的原因，在我国开设计算机操作系统课程的学校里，既能在教学计划中安排操作系统课程教学，又能够安排学生参加系统化的操作系统实习课程的学校并不多。但是这种情况正在逐步开始改变，操作系统实习课程的重要性正在被越来越多的学校所认识。

在我国电子信息类专业教学改革的进程中，学习国外先进经验，重视操作系统实习课程，引进国际一流大学的教材，是非常重要的。不过，考虑到我国多数的普通高等学校的实际情况，简单地照搬国际一流大学操作系统实习课程的内容、大纲和教材，不一定能够取得预想的良好效果。更重要的是，应该设计出符合我国多数高等学校实际情况、具有特色的操作系统实习教材。本书就是在这样的背景之下开始写作的。

本书的作者都是在高等学校里从事操作系统课程教学和相关研究工作的教师。为了使这本操作系统实习教材比较符合我国多数高等学校电子信息类专业的实际情况，在调查研究的基础上，作者们拟订了写作原则和内容大纲，并进行了反复的修改，以期更好地符合实际教学实习的需要。

整个实习课程内容安排的原则如下：

• 实习内容与课程教学彼此呼应

本书在内容上与《Windows操作系统原理》一书相呼应。教材的开始部分安排了Microsoft Windows 2000/XP操作系统原理有关章节的内容，其目的是便于学习，也便于保持本书的相对独立性。

本教程中的有关实习内容围绕着操作系统原理中最重要的基本概念和原理展开。其中第4章的七个小实习——读者写者问题、内存管理、快速文件系统、软盘I/O、Windows Socket网络通信、动态链接库和WDM驱动程序，都是在《Windows操作系统原理》一书中“实习”部分提出的，在本书中进行了详细解答。这七个小实习一般都包括实习目的、实习过程、程序清单以及习题等几个部分。这些内容从各个角度对每个实习课题的目的以及如何进行实习并解决其中的难点做了具体的叙述，而且提供了参考源代码。为了巩固实习的效果，还为这些小实习安排了进一步练习的习题。

学生通过相关实习，可以巩固对原理知识的学习效果，加深对基本概念的理解，并学习如何将基本原理和实际设计有机地结合起来。

• 经典内容和现代发展结合

操作系统的实习内容，要能够有利于学生理解和掌握原理和概念，即所谓经典内容。同时，操作系统的实习要有利于学生掌握当代的操作系统概貌。从这个原则出发，我们把实习内容都安排在 Microsoft Windows 2000/XP上进行。由于Microsoft Windows 2000/XP操作系统的使用极为普及，因此对各个高等学校而言，为实习课程提供Microsoft Windows 2000/XP操作系统环境，应该不会存在很大的困难。

• 掌握基本方法与提高设计水平结合

在安排各个实习课题的内容与实现步骤过程中，我们希望尽可能通过这些实践，帮助读者掌握有关操作系统的基本设计方法，并且切实学习到一些较高水平的设计技巧。本书中有一个微软亚洲研究院提供的实习课题：NDIS协议驱动程序设计（在本书的随书光盘中包括NDIS协议驱动程序设计的部分源代码）。这个课题已经专门安排在第3章中。

一般来说，在Microsoft Windows环境下编写程序的工作，在很多情况下是设计驱动程序，而且往往需要编写支持网络通信的驱动程序。因此，掌握这类驱动程序的设计方法，是很有价值的，这也正是微软亚洲研究院提供这个实习课题的意义所在。

然而，这个实习课题中的程序，不是通常在用户态下运行的程序，这个程序是在操作系统核心态下运行的。读者如果对Windows操作系统内部原理没有一定的理解，很难完成这个实习任务。此外，由于这个程序的编写工作不同于

普通的用户态程序，如果读者不对程序的每一步和每个细节进行仔细的斟酌，那么，即使程序编写完成，在运行这个带有某种bug 的驱动程序时也将会带来整个Windows系统的崩溃！读者如果打算测试和考核一下自己的操作系统理论水平和Windows程序设计能力，那么就尽量不要提前阅读有关的参考内容，而试着去独立完成这个实习课题吧。

• 原理与源代码结合

通常的操作系统原理课程很少涉及有关原理的具体源代码实现。在本书中，为了使学生深入了解操作系统原理和概念的具体实现，对所安排的实习课题一般都提供了参考源代码。提供参考源代码的做法，既便于教师对实习内容的指导工作，也有利于学生们独立学习。这些实习课题都已经在北京大学和清华大学的操作系统实习课上实践过，到目前为止，累计有700多名本科学生参加了实习课程。

不过，限于时间和水平，本书中有关参考源代码的设计不一定是最优的。欢迎读者们提供更好的参考源代码，以便本书再版时改进。

最后，作者对微软亚洲研究院提供了部分操作系统实习课题表示由衷的感谢，感谢微软亚洲研究院大学高校关系部的陈宏刚博士、刘佐扬女士和马歆女士的大力支持。感谢参与编写实习课题参考源代码的北京大学、清华大学和北京航空航天大学的学生，他们是宋翔、李想、王勇、吴墨、钟诚、王会娣、胡建钧、刘晓敏、马梦瑶、顿楠。感谢机械工业出版社华章公司的编辑们，他们对本书的出版给予了有力的支持。最后，对关心本书的所有朋友们表示感谢，他们的理解和支持，是本书出版的保证。

作　者

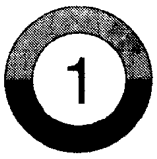
2002年7月18日

目录

CONTENTS

目录

CONTENTS



Windows 2000/XP 操作系统概述

第 1 章

Windows 2000/XP 操作系统概述

作为全书的开篇，本章简要介绍Windows 2000/XP操作系统的体系结构、处理器管理、内存管理、文件管理以及I/O管理的实现机制。如果读者没有读过《Windows操作系统原理》一书，那么通过本章可以尽快地了解Windows 2000/XP操作系统的概貌，为完成本书后续章节的实习打下必要的基础。

1.1 Windows 2000/XP的体系结构

Windows 2000/XP像其他许多操作系统一样，通过硬件机制实现了核心态（管态，kernel mode）和用户态（目态，user mode）两个特权级别，操作系统中那些至关紧要的代码在核心态运行，可以访问系统数据和硬件，而用户程序在用户态运行，不能直接访问操作系统特权代码和数据。这样就使所有操作系统组件都受到了保护，以免被错误的应用程序侵扰，这种保护使得Windows 2000/XP成为相当稳定的工作平台。

Windows 2000/XP体系结构框图如图1-1所示。下面依次介绍构成Windows 2000/XP的各个组成部分体系结构的细节。

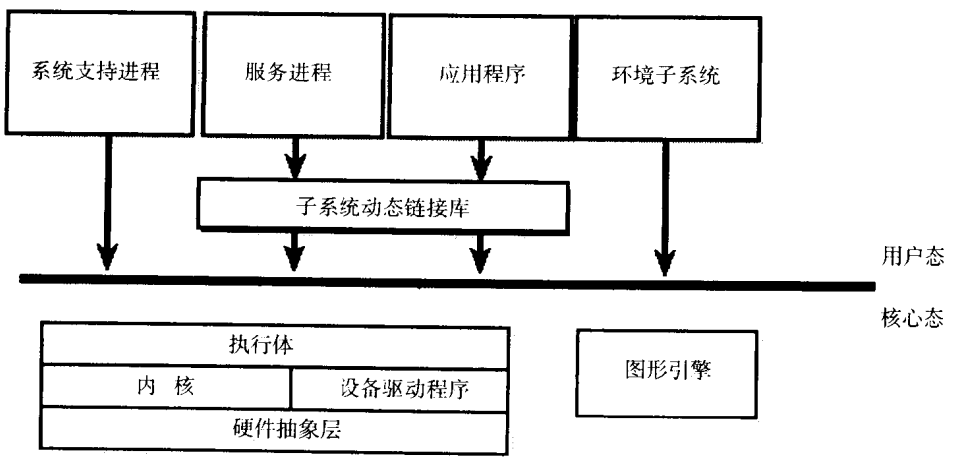


图1-1 Windows 2000/XP体系结构框图

1.1.1 核心态操作系统组件

在Windows 2000/XP中，只有那些对性能影响很大的操作系统组件才在核心态下运行。在核心态下，操作系统组件可以和硬件交互，组件之间也可以交互，并且不会引起上下文切换和模式转变。

可移植性是Windows 2000/XP的一个重要设计目标，即不仅可以在X86体系结构下运行，而且可以在其他硬件平台上运行。为实现这一目标，Windows 2000/XP的核心态操作系统组件采用了分层的结构，将依赖于处理器体系结构或平台的系统底层部分隔离在单独的模块之中，这样系统的高层（执行体，executive）就可以被屏蔽在千差万别的硬件平台之外。提供操作系统可移植性的两个关键组件是硬件抽象层（hardware abstract layer，HAL）和内核（kernel）。依赖于处理器体系结构的功能（如线程上下文切换）在内核中实现，在相同体系结构中，因计算机硬件平台而异的功能在HAL中实现。

1. 硬件抽象层

硬件抽象层是一个可加载的核心态模块HAL.DLL，它为运行Windows 2000/XP的硬件平台提供低层接口，将操作系统从与平台相关的硬件差异中隔离出来。HAL使得每台机器的系统总线、DMA控制器、中断控制器、系统计时器以及多处理器通信机制等对内核来说看上去都是相同的。

2. 内核

内核是ntoskrnl.exe的下层，它实现最基本的操作系统功能，管理线程调度、进程切换、异常和中断处理以及多处理器同步　中断处理、异常调度和多处理器同步等功能是随处理器体系结构的不同而异的，内核的一个重要功能就是把执行体和处理器体系结构的差异隔离开，为执行体提供一组在整个体系结构上可移植的、语义完全相同的接口。

内核是常驻内存的，永远不会由页面调度程序调出内存。与执行体的其他部分和用户应用程序不同，内核自身的代码并不以线程的方式运行。内核可以被中断服务例程（interrupt service routine，ISR）中断，但是永远不会被抢先。

内核除了实现最基本的操作系统功能外，几乎将所有的策略制定留给了执行体。这一点充分体现了Windows 2000/XP将策略与机制分离的设计思想。

3. 执行体

Windows 2000/XP的执行体是ntoskrnl.exe的上层，它由一些重要的系统组件组成，这些组件为用户态的应用程序提供了系统服务功能（即通常所说的本机API）。下面简要描述了主要的执行体组件：

- **对象管理器**：负责创建、跟踪以及删除Windows 2000/XP执行体对象。为对象的命名、维护和安全性设置实施统一的规则。
- **进程与线程管理器**：负责创建、跟踪以及删除进程和线程对象。对进程和线程的基本支持在内核中实现，而执行体的进程与线程管理器给这些低级对象添加附加语义和功能。
- **I/O管理器**：为应用程序提供访问I/O设备的统一框架，负责分发适当的设备驱动程序。
- **安全访问监视器**：为访问受保护对象实施访问确认和审核，受保护对象包括文件、进程、

I/O设备等。

- **本地过程调用（local procedure call，LPC）机制**：以类似于分布式处理中远程过程调用（RPC, Remote Procedure Call）的方式在单机系统中在应用程序和环境子系统之间实现客户/服务器模型。
- **虚拟内存管理器**：负责把进程地址空间中的虚拟地址映射为计算机内存中的物理页面。
- **高速缓存管理器**：通过使最近访问过的磁盘数据驻留在内存中来提供快速访问，从而提高基于文件的I/O性能。

1.1.2 用户进程

图1-1中粗线上部的方框代表了用户进程，它们运行在私有地址空间中。Windows 2000/XP支持四种基本的用户进程，它们是系统支持进程、服务进程、环境子系统和应用程序，下面对它们进行依次介绍。

1. 系统支持进程

系统支持进程是未作为操作系统核心的一部分提供的系统支持服务，例如，登录进程（WINLOGON）和会话管理器（SMSS）。

2. 服务进程

Windows 2000/XP的服务进程类似于UNIX 的守护进程，在客户端/服务器应用程序中扮演服务器角色。Web 服务器就是一个服务进程的例子。

3. 环境子系统

环境子系统向应用程序提供运行环境和应用程序编程接口（Application Programming Interface，API）。Windows 2000/XP支持三种环境子系统：Win32、POSIX和OS/2。Windows 2000/XP最重要的环境子系统是Win32子系统，其他子系统都要通过Win32子系统接收用户的输入和显示输出。

环境子系统的作用是将基本的执行体系统服务的某些子集提供给应用程序。用户应用程序不能直接调用Windows 2000/XP系统服务，这种调用必须通过一个或多个子系统动态链接库作为中介才可以完成。例如，Win32子系统动态链接库（包括kernel32.dll、user32.dll和gd132.dll）实现Win32 API函数。

当一个应用程序调用子系统动态链接库中的函数时，会出现下面三种情况之一：

- 函数完全在子系统动态链接库的用户态部分中实现，这时并没有消息发送到环境子系统进程，也没有调用执行体服务。函数在用户态中执行，结果返回到调用者。
- 函数需要一个或多个对执行体系统服务的调用。
- 函数要求某些工作在环境子系统进程中进行。在这种情况下，将产生一个客户/服务器请求到环境子系统，其中的一个消息将被发送到子系统去执行某些操作，这可能会使用执行体的本地过程调用（LPC）机制。子系统动态链接库在消息返回给调用者之前会一直等待应答。

4. 应用程序

Windows 2000/XP支持五种类型的应用程序：Win32、Windows 3.1、MS-DOS、POSIX 和OS/2。

1.1.3 Windows 2000/XP的对象模型

Windows 2000/XP大量采用了面向对象的概念，简化了进程间资源和数据的共享，便于保护资源免受未经许可的访问。

并非Windows 2000/XP中的所有实体都是对象。当数据或资源对用户态开放时，或者当数据访问是共享的或受限制时，才使用对象。采用对象方法表示的实体有文件、进程、线程、信号量、互斥量、事件、计时器等。Windows 2000/XP通过对象管理器以一致的方法创建和管理所有的对象类型，对象管理器代表应用程序负责创建和删除对象，并负责授权访问对象的数据和服务。

Windows 2000/XP中有两种类型的对象：执行体对象和内核对象。执行体对象是由执行体的各种组件（如进程管理器、内存管理器、I/O管理器等）实现的对象；内核对象是由内核实现的一个更原始的对象集合，内核对象对用户态代码是不可见的，它们仅在执行体内创建和使用。内核对象提供了一些基本性能，许多执行体对象内包含着一个或多个内核对象。

每一个对象都有一个对象头和一个对象体。对象管理器控制对象头，各执行体组件控制它们自己创建的对象类型的对象体。

执行体对象和对象服务都是基本设施，环境子系统用它们来构造自己版本的对象和资源。环境子系统为其应用程序提供的对象集一般与执行体所提供的有些差异。Win32子系统使用执行体对象导出它自己的对象集，其中的大部分直接符合执行体对象。

当进程通过名称来创建或打开一个对象时，它会收到一个代表进程访问对象的句柄。所有用户态进程只有获得了对象句柄之后才可以使用这个对象。句柄作为系统资源的间接指针来使用，这种不直接的方式阻止了应用程序对系统数据结构直接地随便操作。

句柄、执行体对象以及内核对象之间的关系如图1-2所示。

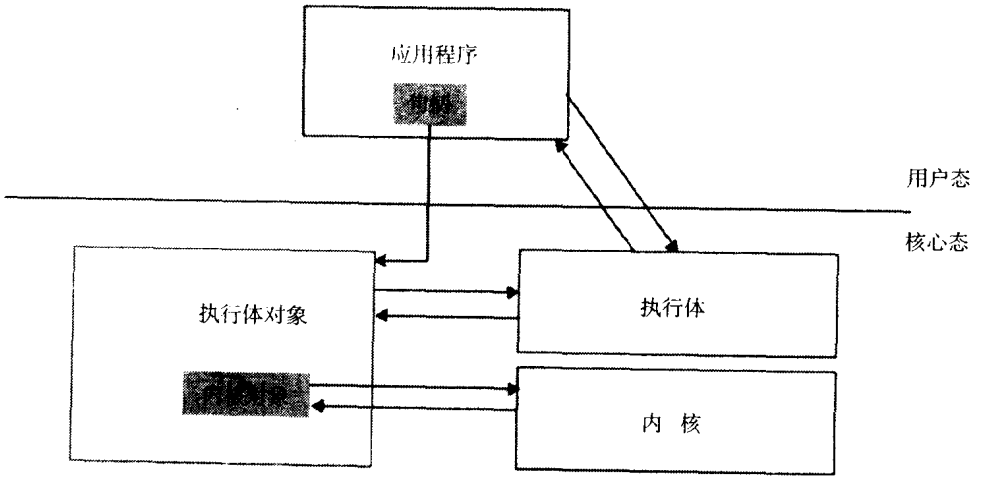


图1-2 句柄、执行体对象与内核对象之间的关系

1.2 Windows 2000/XP的处理器管理

Windows 2000/XP的处理器管理以进程和线程的管理为核心。在Windows 2000/XP中，进程是系统资源分配的基本单位，而线程则是处理器调度的实体。

Windows 2000/XP的进程和线程均被作为对象实现，进程和线程对象体由进程与线程管理器管理，对象头由对象管理器管理，进程和线程对象在内核提供的内核进程对象和线程对象的基础上实现。

1.2.1 Windows 2000/XP中进程的实现

Windows 2000/XP中的每个进程都由一个执行体进程块（EPROCESS）表示，执行体进程块描述进程的基本信息，并指向其他与进程控制相关的数据结构。执行体进程块中的主要内容包括：

- **线程块列表**：描述属于该进程的所有线程的相关信息，以便线程调度器进行处理器资源的分配和回收。
- **虚拟地址描述符(virtual address descriptor，VAD)**：描述进程地址空间各部分属性，用于虚拟存储管理。
- **对象句柄列表**：当进程创建或打开一个对象时，就会得到一个代表该对象的句柄，用于对象访问。对象句柄列表维护该进程正在访问的所有对象列表。

Windows 2000/XP进程结构如图1-3所示。

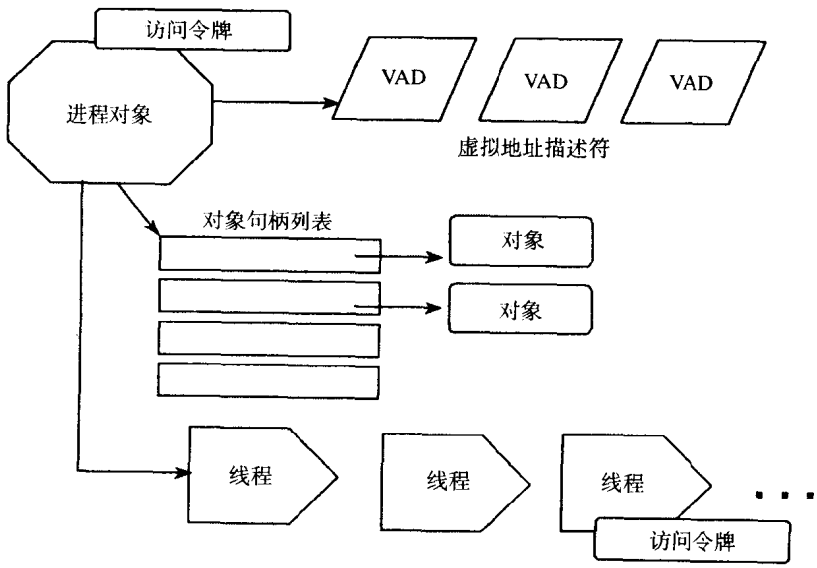


图1-3 Windows 2000/XP的进程结构

Windows 2000/XP支持的各环境子系统都有相应的API函数实现进程控制。Win32子系统的进

程控制API函数主要有CreateProcess、ExitProcess和TerminateProcess。CreateProcess用于进程创建，而ExitProcess和TerminateProcess用于进程退出。这几个API函数的具体使用方法请查阅MSDN文档。

1.2.2 Windows 2000/XP中线程的实现

在Windows 2000/XP中，处理器调度的对象是线程。线程上下文主要包括寄存器、线程环境块、核心栈和用户栈。Windows 2000/XP把线程状态分成七种，如图1-4所示。

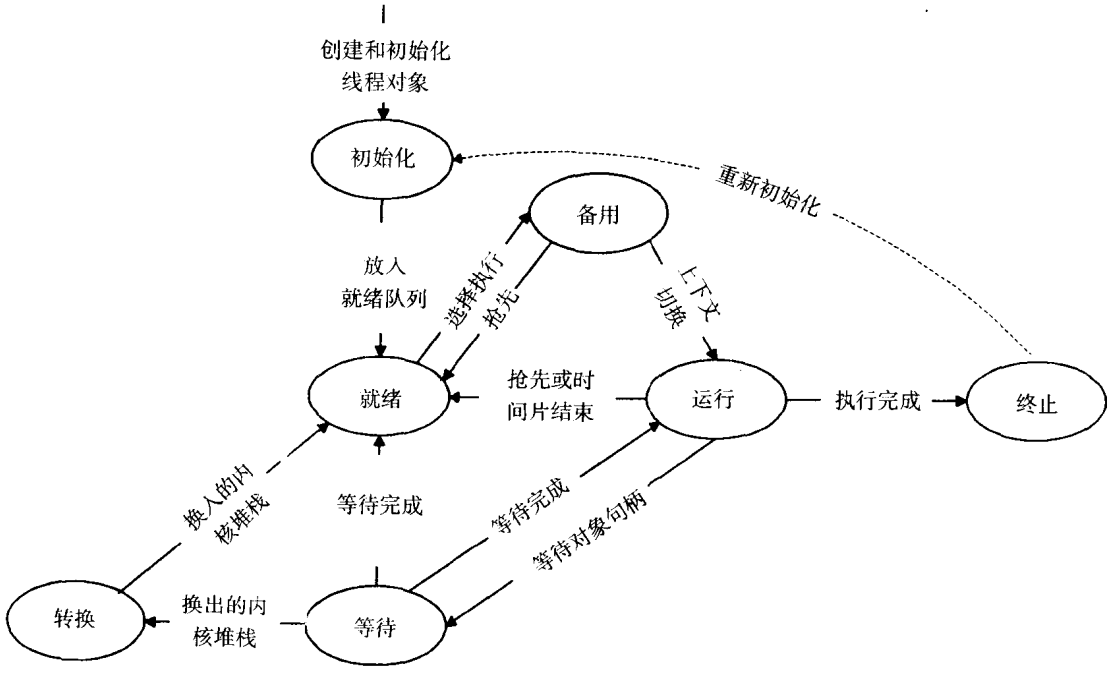


图1-4 Windows 2000/XP的线程状态

- **就绪状态(Ready)**：线程已获得除处理器外的所需资源，正等待调度执行。
- **备用状态(Standby)**：已选择好线程的执行处理器，正等待上下文切换，以进入运行状态。系统中每个处理器上只能有一个处于备用状态的线程。
- **运行状态(Running)**：已完成上下文切换，线程进入运行状态。线程会一直处于运行状态，直到被抢先、时间片用完、线程终止或进入等待状态。
- **等待状态(Waiting)**：线程正等待某对象，以同步线程的执行。当等待事件出现时，等待结束，并根据优先级进入运行或就绪状态。
- **转换状态(Transition)**：转换状态与就绪状态类似，但线程的内核堆栈位于外存。当线程等待事件出现而它的内核堆栈处于外存时，线程进入转换状态；当线程内核堆栈被调回内存