

(第三版)

操作系统基础

屠 祁 屠立德等 编著



清华大学出版社

<http://www.tup.tsinghua.edu.cn>

7.2.6
18.4.1

操作系统基础

(第三版)

屠祁 屠立德 等 编著

清华大学出版社

(京)新登字 158 号

内 容 提 要

本书是一本全面详尽地介绍 Windows NT 和近代 UNIX,并以该系统作为范例的教科书。本书也是一本力求把现代操作系统的典型特征——多线程、微内核、SMP 多处理器系统、分布式系统、客户/服务器模式与经典的操作系统原理紧密结合的教科书。

本书共分 6 部分(14 章)。第 1 部分介绍操作系统的基本概念和运行环境。第 2 部分分别以一章讨论进程和多线程机制以及并行性。第 3 部分主要讨论以 SMP 多处理器调度为中心的处理器管理与死锁。第 4 部分讨论存储管理技术,着重研究了内核主存管理和虚拟存储技术新发展。第 5 部分是设备和文件管理,着重介绍了流和虚拟(多重)文件系统机制。第 6 部分探讨了分布式计算机系统、微内核、操作系统结构和范例。本书作为计算机专业教材,内容丰富,通俗易懂,便于自学。可作为大专院校计算机专业以及研究生的教科书和参考书,也可作为电视大学的教材。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

操作系统基础/屠祁等编著. —北京:清华大学出版社,2000

ISBN 7-302-03943-7

I. 操… II. 屠… III. 操作系统(软件)—教材 IV. IP316

中国版本图书馆 CIP 数据核字(2000)第 32998 号

出版者:清华大学出版社(北京清华大学学研大厦,邮编 100084)

印刷者:北京昌平环球印刷厂

发行者:新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 22.5 字数: 537 千字

版 次: 2000 年 9 月第 3 版 2000 年 9 月第 1 次印刷

书 号: ISBN 7-302-03943-7/TP·2305

印 数: 0001~8000

定 价: 25.50 元

前言

首先我们衷心感谢广大读者的厚爱和支持,本书出版十几年来,已印刷二十多次,发行数十万册。广大读者把他们积累的经验 and 体会无私地贡献给我们,为本书的改进和提高作出了贡献,在此我们向广大读者表示深深的敬意和感谢。

在世纪之交、千禧年之际,我们迎接着新世纪的到来,更期待着我国信息产业的巨大发展。还记得今年仲夏之时,我国计算机界人士一致强烈呼吁“必须迅速发展我国自己的操作系统,它是我国信息产业的坚实基础”。此时,我们修订出版本书第三版,就是本着为推动和发展我国操作系统事业尽我们微薄之力。因此本书宗旨是:致力于经典的操作系统基本原理和概念的深入研讨和阐述;着眼于现代操作系统最新技术的发展和应

用。这次修订的第三版围绕经典操作系统基本原理和概念为框架和基线,以密切反映现代操作系统技术的新发展和新特征为重点。因为几十年来操作系统经历着日新月异的变化,尽管现代操作系统以多线程、微内核、SMP 多处理器系统、客户/服务器模式和分布式、网络系统为特征,但操作系统的基本原理和概念不但没有什么变化,而且更趋成熟,它依然是操作系统课程的基本骨架。但是操作系统在骨架未变的基础上确实有着惊人的变化。那就是:

多线程机制:它已构成了现代操作系统最主要的活动实体。不讲解多线程机制就等于失去操作系统的灵魂。所以本书抽出一章专门介绍多线程,并把多线程概念有机地融合在全书之中,贯穿全书各个章节。

微内核与客户/服务器模式:这两者看似两个不同的概念,但它们一起共同地反映了现代操作系统结构设计的主要趋势。对操作系统的安全性、灵活性、可扩充性等带来强烈而深远的影响。在操作系统课程中不可不加以仔细深入的研究。本书既在专门章节中加以研讨,更将其融合并贯穿全书始终。

多处理器系统:已被广泛使用于工作站和服务器之中,计算机从业人员不可不熟悉它,尤其是对称多处理器(SMP)系统更被广泛使用。本书用专门一章来研究多处理器系统及其调度算法,这也是本次修订的一个重要方面。

分布式系统:它不但是当前发展热点,而且被各团体、单位广为使用,本书也以专门章节对其进行研讨。

对第二版中的许多章节我们也引入了许多新的内容,例如增加了内核主存管理、多重页表和反向页表、流的机制、虚拟或多重文件系统框架等等,可以说每个章节均有较大改动。

本书对国内读者熟悉的 P. V 操作的名称决定在本书中放弃不用,改为 Wait 和 Signal 操作。其原因是这两个操作在程序语言中往往不用 P. V 这种称呼,在国外教科书中也早已不用 P. V 名称。本书中的这种改变,也许会在一开始使教师和读者感到不便,但从长远来说是有好处的。

本书目录中画有“*”号部分,授课教师可以根据教学计划进行变动。但作者以为本书中许多内容也许是“新”的,但在一二年内,这些内容将会是计算机中普遍应该了解掌握的内容,作者鼓励读者通过自学(课内学时有限)来掌握它。本书内容阐述深入浅出,适合自学。

本书中以现代操作系统的典型系统 Windows NT 和近代 UNIX 这两个系统作为全书的范例。对于近代 UNIX 系统如 Mach、Solaris、SVR4 等,是将它们中的先进技术和机制分散到各章中作为该章的典型范例来使用,以收到与全书内容紧密结合的效果,便于读者领会、理解和加深该章所述的内容。而 Windows NT 则作为一个完整的系统范例自成一章,便于读者从系统整体角度来认识和理解一个操作系统。建议读者在阅读各章时也参考 Windows NT 的相关内容。

本书第 1、8、13 章由屠祁编写,第 7、9 章由范群英编写,第 2、3、5 章由屠郑编写,第 12、14 章由郭茵编写,其余部分由屠立德编写。屠祁与屠立德教授负责全书的审阅、校核和定稿。由于时间仓促以及作者水平所限,错误和不妥之处在所难免,恳请读者批评指正。

作 者
1999 年

目 录

第 1 部分 概 论

第 1 章 引论	1
1.1 系统概述	1
1.1.1 计算机的硬件组织	1
1.1.2 软件的层次与虚拟机的概念	3
1.2 操作系统的形成和发展	4
1.2.1 什么是操作系统	4
1.2.2 操作系统的形成和发展	4
1.3 多道程序设计的概念	6
1.3.1 多道程序设计的引入	6
1.3.2 多道程序设计的概念	6
1.4 操作系统的功能和特性	8
1.4.1 操作系统的功能	8
1.4.2 操作系统的特性	10
1.5 操作系统的类型	10
1.5.1 多道批处理操作系统	11
1.5.2 分时系统	11
1.5.3 实时系统	12
1.5.4 网络操作系统	13
1.6 现代操作系统	15
1.6.1 现代操作系统特点	15
1.6.2 Windows NT 简介	16
习题	17
第 2 章 操作系统的运行环境	19
2.1 硬件环境	19
2.1.1 中央处理器(CPU)	19
2.1.2 主存储器	21

2.1.3	缓冲技术	23
2.1.4	中断技术	24
2.1.5	时钟、时钟队列	28
2.2	操作系统与其他系统软件的关系	29
2.2.1	作业、作业步和进程的关系	29
2.2.2	重定位的概念	30
2.2.3	绝对装入程序和相对装入程序	31
2.3	操作系统与人的接口	33
2.3.1	作业控制语言	34
2.3.2	联机作业控制——终端命令和图形用户接口(GUI)	35
*2.4	固件——微程序设计概念	37
2.4.1	微程序设计的概念	37
2.4.2	微程序设计和操作系统	38
习题		39

第2部分 进程、多线程和并行性

第3章	进程管理	41
3.1	进程的概念	41
3.1.1	进程的引入	41
3.1.2	进程的定义	42
3.2	进程的状态	43
3.2.1	进程的状态及其变化	43
3.2.2	进程的挂起和解除挂起的状态	44
3.3	进程的描述和管理	45
3.3.1	进程的描述	45
3.3.2	进程管理	46
3.4	进程控制	47
3.4.1	进程的控制原语	47
3.4.2	操作系统与进程控制的执行	52
3.5	UNIX SVR4 的进程管理	54
习题		57

第4章	多线程	58
4.1	线程的概念	58
4.1.1	线程的引入	58
4.1.2	线程的概念	59
4.2	线程的状态和线程管理	61

4.2.1 线程的状态	61
*4.2.2 线程的描述	62
*4.2.3 线程的管理、线程组	66
*4.3 多线程的实现	67
4.3.1 概说	67
4.3.2 用户级线程	68
4.3.3 内核级线程	70
4.3.4 KLT 和 ULT 结合的方法	71
4.3.5 线程库	71
*4.4 Solaris 操作系统的线程机制	72
4.4.1 Solaris 的多线程结构	72
4.4.2 轻质进程	73
4.4.3 内核线程	74
4.4.4 用户线程	75
4.4.5 线程的执行	75
4.4.6 内核中断线程	76
*4.5 Mach 操作系统的线程机制	77
习题	78

第5章 并行性:互斥和同步	79
5.1 概论	79
5.2 临界段	80
5.2.1 临界段的提出	80
5.2.2 临界段的互斥要求	82
5.3 互斥	82
5.3.1 互斥的软件方法	83
5.3.2 互斥的硬件方法	87
5.4 信号量	90
5.4.1 信号量	90
5.4.2 信号量及同步原语	90
5.4.3 同步原语的不可分割性	92
5.4.4 用信号量实现进程间互斥	93
5.4.5 生产者和消费者问题	94
5.4.6 阅读者和写入者问题	96
5.5 管程	97
5.5.1 管程的定义	97
5.5.2 用管程实现同步	98
5.6 进程间的通信	100

5.6.1	进程通信的实现	101
5.6.2	间接通信模式	102
5.6.3	其他通信模式	102
5.7	UNIX 的进程同步和通信	103
5.7.1	管道(pipes)	103
5.7.2	消息	104
5.7.3	共享主存段	105
5.7.4	信号量	106
5.7.5	信号或软中断	108
5.8	Solaris 线程同步原语	108
	习题	109

第 3 部分 处理器管理、死锁

第 6 章	多处理器管理系统和处理器管理	112
6.1	多处理器系统	112
6.1.1	多处理器系统的优点	112
* 6.1.2	多处理器系统并行性的提高	113
* 6.1.3	多处理器的硬件组织	115
6.1.4	多处理器系统的分类	117
6.1.5	主/从式多处理器系统	117
6.2	对称式多处理器系统(SMP)	118
6.2.1	对称式多处理器系统概念	118
6.2.2	多处理器操作系统	118
6.3	调度的层次和作业调度	120
6.3.1	调度的层次	120
6.3.2	作业状态	121
6.3.3	作业的调度	121
6.4	单处理器系统的处理器调度	122
6.4.1	选择调度算法时应考虑的问题	123
6.4.2	调度算法	124
* 6.5	多处理器系统的处理器管理和调度	128
6.5.1	多处理器调度的概念	128
6.5.2	负载共享调度	130
6.5.3	专用处理器式调度	131
6.5.4	群调度	131
6.5.5	调度类和多模式调度器	132
6.5.6	实时调度	133

* 6.6 UNIX 类系统的处理器调度	135
习题	136
第 7 章 死锁	138
7.1 死锁问题的提出	138
7.2 死锁的必要条件	140
7.2.1 资源的概念	140
7.2.2 死锁的必要条件	140
7.3 死锁的预防	141
7.3.1 预先静态分配法	141
7.3.2 有序资源使用法	142
* 7.4 死锁的避免和银行家算法	142
7.4.1 单资源的银行家算法	143
7.4.2 多资源的银行家算法	144
* 7.5 死锁检测与恢复	145
7.5.1 死锁的检测	145
7.5.2 死锁的恢复	146
习题	146

第 4 部分 主存储器管理

第 8 章 实存储器管理技术	148
8.1 引言	148
8.1.1 主存储器的物理组织、多级存储器	148
8.1.2 主存储器管理功能	149
8.2 固定分区	149
8.3 可变分区多道管理技术	151
8.3.1 可变分区存储管理的概念	151
8.3.2 存储分配算法	152
8.3.3 存储器的紧缩和程序的浮动	154
8.3.4 动态重定位的可变分区多道管理	155
8.4 多重分区(多对界地址)管理	157
8.5 简单分页	157
8.6 简单分段	160
8.7 内核主存管理	161
8.7.1 内核主存管理概述	161
8.7.2 2 次幂空闲表分配器	162
8.7.3 伙伴系统	162

8.7.4	SVR4 的延迟伙伴算法	164
习题		165
第 9 章	虚拟存储管理	166
9.1	虚拟存储系统的基本概念	166
9.2	分页存储管理	167
9.2.1	分页系统中的地址转换	167
9.2.2	硬件支持	171
9.3	分段存储管理	173
9.3.1	分段概述	173
9.3.2	分段的实现	174
9.4	段页式存储管理	175
9.4.1	段页式存储管理的基本概念	175
9.4.2	段页式存储管理中的地址转换	175
9.4.3	段页式存储管理算法	176
9.4.4	段页式存储管理的优缺点	177
9.4.5	Intel Pentium 的段页式机制	178
9.4.6	保护环和调用门	180
9.5	页的置换算法	181
9.5.1	页面访问失效及处理	181
9.5.2	页面置换算法	182
9.5.3	交换区	186
9.6	页架的分配策略	186
9.6.1	物理主存	186
9.6.2	空闲页面链表	187
9.6.3	页架分配中的有关策略	188
9.6.4	分页环境中程序的行为特性	191
9.7	主存共享、快表一致性问题	193
9.7.1	主存共享	193
9.7.2	快表一致性问题	194
9.8	SVR4 UNIX 的存储管理	197
习题		199

第 5 部分 设备和文件管理

第 10 章	设备管理	201
10.1	概述	201
10.2	I/O 子系统的层次模型	202

• VII •

10.2.1	I/O 子系统的设计目标	202
10.2.2	I/O 子系统的层次模型	203
10.3	I/O 硬件组成	204
10.3.1	设备和设备控制器	204
10.3.2	直接存储器访问	205
10.3.3	通道方式和输入输出处理器	206
10.4	设备驱动程序	207
10.4.1	设备和驱动程序分类	207
10.4.2	设备开关表	208
10.4.3	设备驱动程序框架	209
10.5	I/O 子系统	211
10.5.1	设备命名	211
10.5.2	输入输出缓冲区	212
10.5.3	I/O 子系统独立于设备的工作	215
10.6	流	216
10.6.1	流的概念	216
10.6.2	消息和队列	218
10.6.3	流 I/O	220
10.7	磁盘调度	221
10.7.1	磁盘的硬件特性	221
10.7.2	磁盘调度算法	223
10.8	虚拟设备和 SPOOL 系统	226
习题		227
第 11 章	文件系统	229
11.1	文件	230
11.1.1	文件的命名	230
11.1.2	文件的结构	231
11.1.3	文件的类型	232
11.1.4	文件的属性	234
11.1.5	文件的操作	235
11.1.6	文件加锁	235
11.2	目录	236
11.2.1	目录内容	236
11.2.2	文件目录的结构	237
11.2.3	路径名	241
11.2.4	符号连接	242
11.2.5	目录操作	242

11.3 文件系统的实现·····	243
11.3.1 文件空间的分配和管理·····	243
11.3.2 UNIX 系统的目录实现·····	246
11.3.3 磁盘空间的管理·····	248
11.3.4 文件系统在主存的数据结构和打开操作·····	249
11.3.5 文件系统安装·····	251
* 11.4 虚拟文件系统——多重文件系统框架和接口·····	251
11.4.1 vnode/vfs 体系结构的目标和设计思想·····	252
11.4.2 虚拟文件系统接口概述·····	253
11.4.3 安装一个文件系统,虚拟文件系统开关表·····	255
11.5 安全性和保护·····	256
11.5.1 用户确认技术·····	257
11.5.2 保护机制——数据安全性·····	258
11.5.3 其他·····	259
11.5.4 文件的转储和恢复·····	260
习题·····	261

第 6 部分 分布式计算机系统、操作系统结构和范例

* 第 12 章 分布式计算机系统·····	262
12.1 概述·····	262
12.1.1 什么是分布式计算机系统·····	262
12.1.2 分布式系统的优点·····	263
12.2 分布式操作系统特点·····	264
12.2.1 进程通信·····	264
12.2.2 资源管理·····	265
12.2.3 系统结构·····	265
12.3 进程通信·····	266
12.3.1 进程通信概述·····	266
12.3.2 TCP/IP 通信协议·····	267
12.3.3 分布式环境的客户/服务器模式·····	270
12.3.4 分布式进程通信·····	272
12.4 分布式文件系统·····	277
12.4.1 分布式文件系统概述·····	277
12.4.2 分布式文件系统的组成·····	278
12.4.3 分布式文件系统的体系结构·····	279
12.4.4 客户端高速缓存和一致性·····	282
12.5 分布式系统中的互斥与死锁·····	283

12.5.1	逻辑钟和逻辑时	283
12.5.2	时间戳算法(Lamport 算法)	284
12.5.3	令牌传送算法	285
12.6	进程迁移	287
12.6.1	进程迁移的原因	287
12.6.2	进程迁移机制	288
习题		289
第 13 章	微内核、操作系统的结构和设计	290
13.1	微内核	290
13.1.1	使用微内核结构的优点	290
13.1.2	微内核结构	292
13.1.3	微内核的实现	293
13.2	操作系统的设计	294
13.2.1	设计的目标和原则	294
13.2.2	操作系统的设计	296
13.3	操作系统的结构	298
13.3.1	模块接口法(单块式)	298
13.3.2	层次结构设计法	299
13.3.3	客户/服务器方式	300
习题		301
* 第 14 章	Windows NT 操作系统	303
14.1	Windows NT 操作系统概述	303
14.2	Windows NT 的设计目标	304
14.3	Windows NT 的系统模型	305
14.4	Windows NT 的结构	307
14.4.1	NT 的保护子系统	307
14.4.2	NT 执行体	308
14.4.3	客户/服务器模型实现的例子	309
14.4.4	关于 NT 的结构	310
14.5	Windows NT 的基元成分——对象、进程和线程	310
14.5.1	对象	310
14.5.2	进程	313
14.5.3	线程	315
14.5.4	进程管理程序	317
14.6	微内核和对称多处理器系统	318
14.6.1	微内核和对称多处理器系统(SMP)	318

14.6.2	NT 的线程状态转换	318
14.6.3	内核调度程序	319
14.6.4	进程和线程的优先级	322
14.7	NT 的同步对象	323
14.7.1	线程同步概述	323
14.7.2	用 NT 对象进行同步	324
14.8	虚拟存储管理	326
14.8.1	进程的虚拟地址空间	326
14.8.2	NT 的虚拟分页	327
14.8.3	页面调度策略和工作集	330
14.8.4	页架状态和页架数据结构	331
14.8.5	主存映射文件和视图	332
14.9	输入输出系统	334
14.9.1	输入输出(I/O)系统的结构	335
14.9.2	统一的驱动程序模型	336
14.9.3	异步 I/O 操作和 I/O 请求处理过程	337
14.9.4	映射文件 I/O	337
14.10	Windows NT 的内装网络	337
14.10.1	Windows NT 的内装网络的特色	338
14.10.2	Windows NT 网络的体系结构	339
14.11	对象管理程序	340
14.12	进程通信——本地过程调用(LPC)	341
14.13	Windows NT 的安全性	341
14.13.1	NT 安全性	341
14.13.2	存取令牌和安全描体	342
14.14	综述	344
	习题	344
	参考文献	346

第 1 部分

概 论

第 1 章 引 论

1.1 系统概述

现在的一个完整的计算机系统,不论是大型机、小型机、甚至微型机,都由两大部分组成:即计算机的硬件部分和计算机的软件部分。通常计算机的硬件部分是指计算机物理装置本身,它可以是电子的、电的、磁的、机械的、光的元件或装置,或者是由它们组成的计算机部件和计算机本体。总而言之,硬件部分是指计算机系统中所有那些“硬的”物理设施,也就是指计算机的各种处理器(如中央处理器,输入输出处理器和包含在该计算机系统内的其他处理器)、存储器、输入输出设备和通信装置。而软件部分是指计算机系统内的所有软件。术语“软件”是相对于硬件而言的,它是指由计算机硬件执行以完成一定任务的所有程序及其数据。计算机的软件部分通常指各种语言的编译程序和解释程序、汇编程序、装入程序、连接编辑程序、连接装入程序、用户应用程序、数据库管理系统、数据通信系统和操作系统。那么计算机系统的各硬件部分是怎样连接和构成一个完整的计算机,各软件部分又是怎样的关系,硬件和软件之间又是怎样的关系呢?

1.1.1 计算机的硬件组织

计算机的基本组成如图 1.1 所示,通常也称为冯·诺伊曼结构,它由五部分组成:主机部分由运算器、控制器、存储器组成,外设部分由输入设备和输出设备组成。原始的诺伊曼机在结构上是以运算器为中心的。随着计算机体系结构的发展,而逐渐演变到现在以存储器系统为中心。

计算机部件之间以总线相联接。所谓总线实际上是一组并行的导线,导线数目和计算机字长相同,数据和指令通过总线传送。图 1.2 是以存储器为中心的双总线结构。

至今为止,计算机的发展已经历了四代。第一代是电子管计算机,第二代晶体管计算机,第三代集成电路计算机,第四代大规模集成电路计算机,目前各国正在发展所谓第五

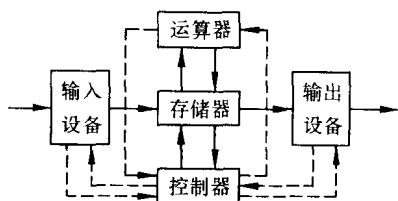


图 1.1 计算机的基本组成

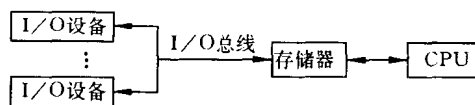


图 1.2 以存储器为中心的双总线结构

代计算机。在这个过程中,计算机性能指标的提高,总是伴随着系统结构的新发展。早期,由于硬件价格过高,设计硬件时,往往把许多工作留给软件去做,这导致了软件的复杂、价格昂贵,以致引起了软件危机。现在硬件已非常便宜,所以计算机结构的发展,在增强功能前提下,更注重其处理效率,因而并行性概念成为推动计算机系统结构发展的重要因素。这包括使原有部件尽可能并行运行。在这方面具有重要意义的进展是通道的出现。通道实际上是一个专用的输入输出 I/O 处理器,由它负责和控制 I/O 设备的工作,从而大大节省了 CPU 机时,提高了 CPU 效率,图 1.3 所示为一个典型的以存储器为中心,具有通道的双总线结构。并行性另一方面努力往往是把一件工作分为若干个相互联系的部分(如把指令的执行工作分解为取指、译码、取数、运算、存结果等),每一部分由专门的处理部件来完成,然后是按流水线原则把各部分的执行并行进行的流水处理器。如果把处理部件进一步发展到处理器,就构成流水线多处理器系统(如把编译过程的扫描、分析、代码生成等部分,都设立专门的处理器,并与执行机器语言的通用处理器相连,就构成了高级语言处理器)。

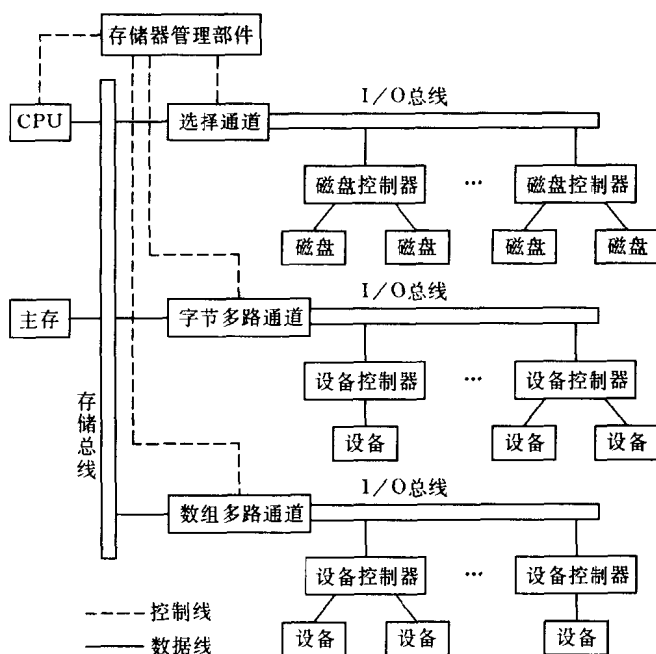


图 1.3 IBM 4300 系统结构