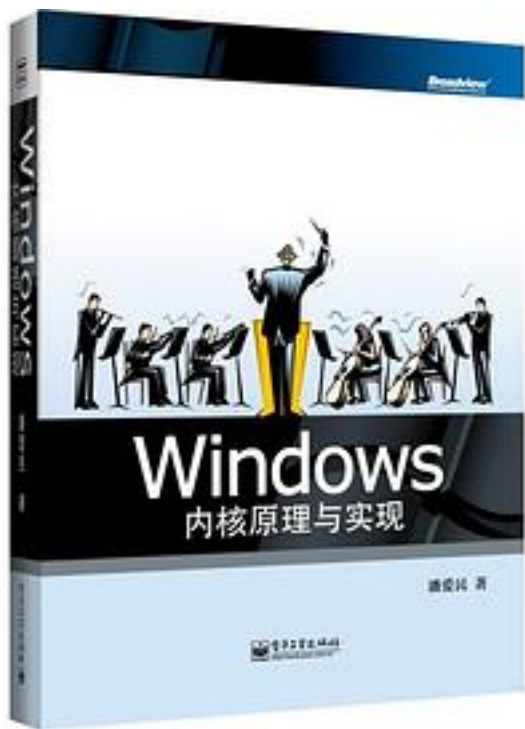


# Windows内核原理与实现



[Windows内核原理与实现 下载链接1](#)

著者:潘爱民

出版者:电子工业出版社

出版时间:2013-5

装帧:平装

isbn:9787121200564

本书介绍Windows内核的基本原理，包括进程和线程、内存管理、线程间同步、I/O模型和Windows的存储模型。对于每一部分内容的介绍，首先从现代操作系统的基本原理出发，然后结合Windows公开的源代码WRK来介绍Windows中的具体实现，最后介绍相应的工具来检查所学的知识。

作者介绍:

潘爱民

任职于阿里云计算有限公司，担任阿里云OS首席架构师。长期从事软件和系统技术的研究与开发工作，撰写了大量软件技术文章，著译了多部经典计算机图书，在国内外学术刊物上发表了30多篇文章。曾经任教于北京大学和清华大学（兼职）。后进入工业界，先后任职于微软亚洲研究院、盛大网络发展有限公司和阿里云计算有限公司。目前也是工信部移动操作系统专家组成员。

潘爱民先生获得了数学学士学位和计算机科学博士学位，主要研究领域包括软件设计、信息安全、操作系统和互联网技术。

## 目录: 第1章 概述 1

### 1.1 操作系统基础 2

#### 1.1.1 计算机系统的硬件资源管理 2

#### 1.1.2 为应用程序提供执行环境 5

### 1.2 学习操作系统之必备知识 7

### 1.3 Windows操作系统发展历史 9

### 1.4 Windows内核的版本 11

### 1.5 操作系统的研究与发展 13

### 1.6 本章总结 16

## 第2章 Windows系统总述 17

### 2.1 现代操作系统的基本结构 17

### 2.2 Windows系统结构 18

#### 2.2.1 Windows内核结构 20

#### 2.2.2 Windows内核中的关键组件 22

#### 2.2.3 Windows子系统 32

#### 2.2.4 系统线程和系统进程 35

### 2.3 关于Windows研究内核 37

#### 2.3.1 WRK包含了什么 38

#### 2.3.2 WRK源代码说明 39

#### 2.3.3 本书对WRK源代码的引用 41

### 2.4 Windows内核的基本概念 42

#### 2.4.1 处理器模式 43

#### 2.4.2 内存管理 44

#### 2.4.3 进程和线程管理 46

#### 2.4.4 中断和异常 48

#### 2.4.5 同步 51

### 2.5 Windows内核中的公共管理设施 53

#### 2.5.1 Windows内核中的对象管理 53

#### 2.5.2 注册表和配置管理器 61

#### 2.5.3 事件追踪（ETW） 72

#### 2.5.4 安全性管理 75

### 2.6 Windows引导过程 81

#### 2.6.1 内核加载 82

#### 2.6.2 内核初始化 85

#### 2.6.3 建立用户登录会话 90

### 2.7 本章总结 96

## 第3章 Windows进程和线程 97

### 3.1 进程基本概念 97

#### 3.1.1 多进程模型 98

#### 3.1.2 进程与程序 99

### 3.2 线程基本概念 102

#### 3.2.1 线程模型 102

#### 3.2.2 线程调度算法 104

3.2.3 线程与进程的关系	106
3.3 Windows中进程和线程的数据结构	106
3.3.1 内核层的进程和线程对象	106
3.3.2 执行体层的进程和线程对象	118
3.4 Windows的进程和线程管理	129
3.4.1 Windows进程的句柄表	129
3.4.2 获得当前线程或进程	135
3.4.3 进程和线程的创建过程	136
3.4.4 进程和线程的结束处理	146
3.4.5 系统初始进程和线程	148
3.5 Windows中的线程调度	150
3.5.1 线程优先级	150
3.5.2 线程状态转移	153
3.5.3 时限管理	163
3.5.4 优先级调度和环境切换	165
3.6 进程和线程运行状态监视工具	171
3.6.1 ProcMon使用示例	171
3.6.2 ProcMon实现原理	173
3.7 本章总结	174
第4章 Windows内存管理	175
4.1 内存管理概述	176
4.1.1 页式内存管理	177
4.1.2 段式内存管理	181
4.1.3 内存管理算法介绍	184
4.1.4 Windows内存管理概述	192
4.2 Windows系统内存管理	194
4.2.1 系统地址空间初始化	194
4.2.2 系统地址空间内存管理	209
4.2.3 系统PTE区域的管理	223
4.3 进程内存管理	229
4.3.1 地址空间的创建和初始化	229
4.3.2 地址空间切换	234
4.3.3 进程地址空间的内存管理	235
4.3.4 内存区对象	241
4.4 内存页面交换	250
4.4.1 Intel x86中的PTE	251
4.4.2 软件PTE：无效PTE和原型PTE	253
4.4.3 页面错误处理	257
4.4.4 Windows的写时复制	263
4.5 物理内存管理	265
4.5.1 PFN数据库	266
4.5.2 物理页面的状态变化	272
4.5.3 物理页面链表的管理和操作	275
4.5.4 修改页面写出器	280
4.5.5 进程/栈交换器	282
4.5.6 低内存通知和高内存通知	285
4.6 工作集管理	286
4.6.1 Windows工作集管理器	286
4.6.2 平衡集管理器	292
4.7 内存监视工具MemMon	293
4.7.1 MemMon使用介绍	293
4.7.2 MemMon实现原理	295
4.8 本章总结	295
第5章 Windows并发和同步	297

5.1 进程和线程的同步基础	297
5.1.1 并发性基础	298
5.1.2 进程或线程之间的通信	301
5.1.3 经典的同步问题	305
5.2 Windows中断与异常	310
5.2.1 硬件中断的发生和处理	311
5.2.2 中断请求级别 (IRQL)	317
5.2.3 中断对象	320
5.2.4 DPC (延迟过程调用)	323
5.2.5 时钟中断和定时器管理	327
5.2.6 APC (异步过程调用)	330
5.2.7 异常分发	336
5.3 不依赖于线程调度的同步机制	343
5.3.1 提升IRQL实现数据同步	343
5.3.2 互锁操作	345
5.3.3 无锁的单链表实现	346
5.3.4 自旋锁	349
5.4 基于线程调度的同步机制	354
5.4.1 线程进入等待	354
5.4.2 分发器对象	361
5.4.3 门等待	369
5.4.4 执行体资源 (executive resource)	370
5.4.5 推锁 (push lock)	373
5.4.6 死锁	378
5.5 观察线程同步关系——DPerfLite	379
5.5.1 DPerfLite使用示例	379
5.5.2 DPerfLite实现原理	381
5.6 本章总结	382
第6章 Windows I/O系统	383
6.1 I/O概述	384
6.1.1 现代计算机系统的I/O	384
6.1.2 I/O软件技术	388
6.1.3 Windows I/O系统结构	390
6.2 I/O管理器	392
6.2.1 驱动程序初始化	393
6.2.2 驱动程序对象和设备对象	399
6.2.3 文件对象	404
6.2.4 对象生命周期管理	407
6.3 即插即用管理器	408
6.3.1 即插即用的基本要求	409
6.3.2 Windows中驱动程序的即插即用支持	410
6.3.3 设备列举与设备树	411
6.4 电源管理器	414
6.4.1 电源管理概述	414
6.4.2 Windows中的电源管理	417
6.5 设备驱动程序	422
6.5.1 设备驱动程序分类	423
6.5.2 例子驱动程序toaster	425
6.5.3 驱动程序的代码结构	427
6.5.4 toaster设备的设备栈	432
6.5.5 过滤驱动程序的配置和加载	434
6.5.6 非即插即用驱动程序	437
6.6 I/O处理	440
6.6.1 I/O请求包 (IRP)	440

6.6.2 针对独立设备对象的I/O处理	447
6.6.3 处理I/O请求过程中的事项	451
6.6.4 针对设备栈的I/O处理	461
6.6.5 I/O完成端口	465
6.7 I/O请求监视工具IRPMon	468
6.7.1 IRPMon使用介绍	468
6.7.2 IRPMon实现原理	469
6.8 本章总结	470
第7章 Windows存储管理	471
7.1 存储管理概述	471
7.1.1 硬件存储体系 (memory hierarchy)	472
7.1.2 Windows的存储管理结构	474
7.2 Windows缓存管理	476
7.2.1 Windows缓存空间的内存管理	476
7.2.2 缓存管理器的数据访问路径	483
7.2.3 直接使用系统缓存中的数据	486
7.2.4 缓存管理器的预读处理	493
7.2.5 缓存管理器的延迟写	496
7.3 Windows中卷的管理	499
7.3.1 Windows存储栈结构	500
7.3.2 卷的挂载	504
7.3.3 卷与文件系统	507
7.3.4 文件对象的I/O处理	510
7.4 Windows文件系统	513
7.4.1 文件系统驱动程序结构	514
7.4.2 RAW文件系统与FsRtl	519
7.4.3 文件系统的I/O过滤	522
7.4.4 FAT文件系统	530
7.4.5 NTFS文件系统	534
7.5 本章总结	542
第8章 Windows系统服务	545
8.1 Windows系统服务原理	545
8.1.1 Intel x86的用户模式-内核模式切换	545
8.1.2 Windows的用户模式-内核模式切换	550
8.1.3 Windows中的系统服务分发	555
8.1.4 增加系统服务表或表项	562
8.2 LPC (本地过程调用) 服务	565
8.2.1 LPC结构模型	565
8.2.2 LPC端口和LPC消息	567
8.2.3 LPC通信模型的实现	569
8.2.4 LPC应用	575
8.3 命名管道 (Named Pipe) 服务	577
8.3.1 命名管道的名称解析	577
8.3.2 命名管道的通信模型	579
8.3.3 命名管道的实现	581
8.4 邮件槽 (Mailslot) 服务	584
8.4.1 邮件槽的名称解析	584
8.4.2 邮件槽的通信模型	585
8.4.3 邮件槽的实现	586
8.5 SDT显示工具SDTViewer	588
8.5.1 SDTViewer使用介绍	588
8.5.2 SDTViewer实现原理	589
8.6 本章总结	590
第9章 Windows系统高级话题	591

9.1 网络	591
9.1.1 Windows网络体系结构	591
9.1.2 TDI（传输驱动程序接口）	595
9.1.3 NDIS（网络驱动程序接口规范）	599
9.1.4 Windows Vista及以后版本的网络结构	601
9.2 Windows子系统	603
9.2.1 Windows子系统结构	603
9.2.2 Windows子系统初始化与GUI线程	607
9.2.3 窗口管理	610
9.2.4 GDI（图形设备接口）	620
9.2.5 Windows Vista及以后的子系统变化	627
9.3 内核日志	629
9.3.1 内核日志记录器	629
9.3.2 利用内核日志信息诊断性能问题	632
9.4 Windows Vista/Server 2008/7的重要变化	640
9.4.1 MinWin工程	640
9.4.2 进程和线程管理	643
9.4.3 内存管理	645
9.4.4 I/O处理的改进	647
9.5 本章总结	650
附录A 建立WRK工作环境	651
A.1 编译WRK	651
A.2 启动WRK	655
A.3 调试WRK	658
附录B 内核代码插入工具KlInjectToolKit	665
B.1 KlInjectToolKit功能介绍	666
B.2 KlInjectToolKit的代码实现	667
B.3 KlInjectToolKit的限制	671
参考资料	673
术语对照表	681
索引	687
• • • • •	<a href="#">(收起)</a>

[Windows内核原理与实现\\_下载链接1](#)

## 标签

计算机科学

Windows内核

操作系统

软件工程

计算机

[技术]操作系统

[Windows]

Windows

## 评论

有点难

-----  
[Windows内核原理与实现 下载链接1](#)

## 书评

英文名：Understanding the Windows Kernel 作者：潘爱民 第1章 概述  
没有太重要需要记录的东西，就是重新回顾一下操作系统特别是win系列的发展。后面每一章都很长很多，需要做好准备，尤其下一章介绍如何配合wrk学习的一节，需要认真学习 第2章 Windows系统概述 2....

-----  
[http://blog.sina.com.cn/s/blog\\_4caedc7a0100k8jt.html](http://blog.sina.com.cn/s/blog_4caedc7a0100k8jt.html)  
在微软工作，最有吸引力的地方是能够融入微软的大家庭中，并触摸到方方面面的技术和产品。微软的产品线遍布软件技术的各个方向，真正称得上软件帝国。对于软件技术人员，这是极好的机会来满足自己的求知欲， ...

-----  
读了这本书，虽然我很想对一个东西刨根问底，但是突然觉得读了没有太大的用处，不过了解底层细节的话，写东西的时候会更有把握一点，对于一些设计的方法也可以简单的借鉴，但是其他的用途，不是那么容易就能表现出来

