

目 录

第一章 走进 Linux	1
1.1 GNU 与 Linux 的成长	1
1.2 Linux 的开发模式和运作机制	2
1.3 走进 Linux 内核	4
1.3.1 Linux 内核的特征	4
1.3.2 Linux 内核版本的变化	5
1.4 分析 Linux 内核的意义	7
1.4.1 开发适合自己的操作系统	8
1.4.2 开发高水平软件	9
1.4.3 有助于计算机科学的教学和科研	9
1.5 Linux 内核结构	9
1.5.1 Linux 内核在整个操作系统中的位置	10
1.5.2 Linux 内核的作用	11
1.5.3 Linux 内核的抽象结构	11
1.6 Linux 内核源代码	12
1.6.1 多版本的内核源代码	13
1.6.2 Linux 内核源代码的结构	13
1.6.3 从何处开始阅读源代码	14
1.7 Linux 内核源代码分析工具	16
1.7.1 Linux 超文本交叉代码检索工具	16
1.7.2 Windows 平台下的源代码阅读工具 (Source Insight)	17
第二章 Linux 运行的硬件基础	19
2.1 i386 的寄存器	19
2.1.1 通用寄存器	19
2.1.2 段寄存器	20
2.1.3 状态和控制寄存器	20
2.1.4 系统地址寄存器	23
2.1.5 调试寄存器和测试寄存器	24
2.2 内存地址	25
2.3 段机制和描述符	26
2.3.1 段机制	26

2.3.2	描述符的概念	27
2.3.3	系统段描述符	29
2.3.4	描述符表	30
2.3.5	选择符与描述符表寄存器	30
2.3.6	描述符投影寄存器	32
2.3.7	Linux 中的段	32
2.4	分页机制	34
2.4.1	分页机构	36
2.4.2	页面高速缓存	39
2.5	Linux 中的分页机制	40
2.5.1	与页相关的数据结构及宏的定义	41
2.5.2	对页目录及页表的处理	42
2.6	Linux 中的汇编语言	44
2.6.1	AT&T 与 Intel 汇编语言的比较	44
2.6.2	AT&T 汇编语言的相关知识	46
2.6.3	gcc 嵌入式汇编	49
2.6.4	Intel386 汇编指令摘要	52
第三章	中断机制	55
3.1	中断基本知识	55
3.1.1	中断向量	55
3.1.2	外设可屏蔽中断	56
3.1.3	异常及非屏蔽中断	57
3.1.4	中断描述符表	59
3.1.5	相关汇编指令	60
3.2	中断描述符表的初始化	61
3.2.1	外部中断向量的设置	61
3.2.2	中断描述符表 IDT 的预初始化	63
3.2.3	中断向量表的最终初始化	65
3.3	异常处理	68
3.3.1	在内核栈中保存寄存器的值	68
3.3.2	中断请求队列的初始化	70
3.3.3	中断请求队列的数据结构	70
3.4	中断处理	77
3.4.1	中断和异常处理的硬件处理。	77
3.4.2	Linux 对中断的处理	78
3.4.3	与堆栈有关的常量、数据结构及宏	79
3.4.4	中断处理程序的执行	81
3.4.5	从中断返回	85

3.5 中断的后半部分处理机制	86
3.5.1 为什么把中断分为两部分来处理	86
3.5.2 实现机制	87
3.5.3 数据结构的定义	89
3.5.4 软中断、bh 及 tasklet 的初始化	91
3.5.5 后半部分的执行	92
3.5.6 把 bh 移植到 tasklet	96
第四章 进程描述	97
4.1 进程和程序 (Process and Program)	97
4.2 Linux 中的进程概述	99
4.3 task_struct 结构描述	100
4.4 task_struct 结构在内存中的存放	107
4.4.1 进程内核栈	107
4.4.2 当前进程 (current 宏)	108
4.5 进程组织方式	109
4.5.1 哈希表	109
4.5.2 双向循环链表	110
4.5.3 运行队列	111
4.5.4 进程的运行队列链表	111
4.5.5 等待队列	112
4.6 内核线程	115
4.7 进程的权能	116
4.8 内核同步	117
4.8.1 信号量	118
4.8.2 原子操作	118
4.8.3 自旋锁、读写自旋锁和大读者自旋锁	119
第五章 进程调度与切换	123
5.1 Linux 时间系统	123
5.1.1 时钟硬件	123
5.1.2 时钟运作机制	124
5.1.3 Linux 时间基准	125
5.1.4 Linux 的时间系统	126
5.2 时钟中断	126
5.2.1 时钟中断的产生	126
5.2.2 Linux 实现时钟中断的全过程	127
5.3 Linux 的调度程序——Schedule ()	131
5.3.1 基本原理	132

5.3.2	Linux 进程调度时机	133
5.3.3	进程调度的依据	135
5.3.4	进程可运行程度的衡量	136
5.3.5	进程调度的实现	137
5.4	进程切换	139
5.4.1	硬件支持	139
5.4.2	进程切换	142
第六章	Linux 内存管理	147
6.1	Linux 的内存管理概述	147
6.1.1	Linux 虚拟内存的实现结构	148
6.1.2	内核空间和用户空间	149
6.1.3	虚拟内存实现机制间的关系	151
6.2	Linux 内存管理的初始化	152
6.2.1	启用分页机制	152
6.2.2	物理内存的探测	157
6.2.3	物理内存的描述	163
6.2.4	页面管理机制的初步建立	166
6.2.5	页表的建立	173
6.2.6	内存管理区	177
6.3	内存的分配和回收	185
6.3.1	伙伴算法	186
6.3.2	物理页面的分配和释放	187
6.3.3	Slab 分配机制	194
6.3.4	内核空间非连续内存区的管理	201
6.4	地址映射机制	204
6.4.1	描述虚拟空间的数据结构	205
6.4.2	进程的虚拟空间	209
6.4.3	内存映射	212
6.5	请页机制	218
6.5.1	页故障的产生	218
6.5.2	页错误的定位	219
6.5.3	进程地址空间中的缺页异常处理	220
6.5.4	请求调页	221
6.5.5	写时复制	223
6.5.6	对本节的几点说明	225
6.6	交换机制	225
6.6.1	交换的基本原理	225
6.6.2	页面交换守护进程 kswapd	229

6.6.3	交换空间的数据结构	233
6.6.4	交换空间的应用	234
6.7	缓存和刷新机制	236
6.7.1	Linux 使用的缓存	236
6.7.2	缓冲区高速缓存	237
6.7.3	翻译后援存储器(TLB)	240
6.7.4	刷新机制	242
6.8	进程的创建和执行	245
6.8.1	进程的创建	245
6.8.2	程序执行	252
6.8.3	执行函数	255
第七章	进程间通信	263
7.1	管道	263
7.1.1	Linux 管道的实现机制	264
7.1.2	管道的应用	265
7.1.3	命名管道 CFIFO	267
7.2	信号 (signal)	267
7.2.1	信号种类	268
7.2.2	信号掩码	270
7.2.3	系统调用	271
7.2.4	典型系统调用的实现	272
7.2.5	进程与信号的关系	274
7.2.6	信号举例	275
7.3	System V 的 IPC 机制	276
7.3.1	信号量	276
7.3.2	消息队列	282
7.3.3	共享内存	285
第八章	虚拟文件系统	289
8.1	概述	289
8.2	VFS 中的数据结构	292
8.2.1	超级块	292
8.2.2	VFS 的索引节点	295
8.2.3	目录项对象	297
8.2.4	与进程相关的文件结构	298
8.2.5	主要数据结构间的关系	302
8.2.6	有关操作的数据结构	302
8.3	高速缓存	308

8.3.1	块高速缓存	308
8.3.2	索引节点高速缓存	312
8.3.3	目录高速缓存	315
8.4	文件系统的注册、安装与卸载	316
8.4.1	文件系统的注册	316
8.4.2	文件系统的安装	319
8.4.3	文件系统的卸载	326
8.5	限额机制	326
8.6	具体文件系统举例	328
8.6.1	管道文件系统 pipefs	329
8.6.2	磁盘文件系统 BFS	332
8.7	文件系统的系统调用	333
8.7.1	open 系统调用	333
8.7.2	read 系统调用	335
8.7.3	fcntl 系统调用	336
8.8	Linux 2.4 文件系统的移植问题	337
第九章	Ext2 文件系统	343
9.1	基本概念	343
9.2	Ext2 的磁盘布局和数据结构	345
9.2.1	Ext2 的磁盘布局	345
9.2.2	Ext2 的超级块	346
9.2.3	Ext2 的索引节点	349
9.2.4	组描述符	352
9.2.5	位图	353
9.2.6	索引节点表及实例分析	353
9.2.7	Ext2 的目录项及文件的定位	358
9.3	文件的访问权限和安全	361
9.4	链接文件	363
9.5	分配策略	366
9.5.1	数据块寻址	367
9.5.2	文件的洞	368
9.5.3	分配一个数据块	369
第十章	模块机制	373
10.1	概述	373
10.1.1	什么是模块	373
10.1.2	为什么要使用模块?	374
10.1.3	Linux 内核模块的优缺点	374

10.2	实现机制	375
10.2.1	数据结构	375
10.2.2	实现机制的分析	379
10.3	模块的装入和卸载	385
10.3.1	实现机制	385
10.3.2	如何插入和卸载模块	386
10.4	内核版本	387
10.4.1	内核版本与模块版本的兼容性	387
10.4.2	从版本 2.0 到 2.2 内核 API 的变化	388
10.4.3	把内核 2.2 移植到内核 2.4	392
10.5	编写内核模块	400
10.5.1	简单内核模块的编写	401
10.5.2	内核模块的 Makefiles 文件	401
10.5.3	内核模块的多个文件	402
第十一章	设备驱动程序	405
11.1	概述	405
11.1.1	I/O 软件	405
11.1.2	设备驱动程序	407
11.2	设备驱动基础	409
11.2.1	I/O 端口	409
11.2.2	I/O 接口及设备控制器	410
11.2.3	设备文件	411
11.2.4	VFS 对设备文件的处理	413
11.2.5	中断处理	413
11.2.6	驱动 DMA 工作	416
11.2.7	I/O 空间的映射	417
11.2.8	设备驱动程序框架	419
11.3	块设备驱动程序	420
11.3.1	块设备驱动程序的注册	420
11.3.2	块设备基于缓冲区的数据交换	422
11.3.3	块设备驱动程序的几个函数	428
11.3.4	RAM 盘驱动程序的实现	432
11.3.5	硬盘驱动程序的实现	433
11.4	字符设备驱动程序	437
11.4.1	简单字符设备驱动程序	437
11.4.2	字符设备驱动程序的注册	438
11.4.3	一个字符设备驱动程序的实例	440
11.4.4	驱动程序的编译与装载	445

第十二章 网络	447
12.1 概述	447
12.2 网络协议	448
12.2.1 网络参考模型	448
12.2.2 TCP/IP 工作原理及数据流	449
12.2.3 Internet 协议	449
12.2.4 TCP	450
12.3 套接字 (socket)	452
12.3.1 套接字在网络中的地位和作用	452
12.3.2 套接字接口的种类	453
12.3.3 套接字的工作原理	454
12.3.4 socket 的通信过程	456
12.3.5 socket 为用户提供的系统调用	460
12.4 套接字缓冲区 (sk_buff)	461
12.4.1 套接字缓冲区的特点	461
12.4.2 套接字缓冲区操作基本原理	461
12.4.3 sk_buff 数据结构的核心内容	463
12.4.4 套接字缓冲区提供的函数	465
12.4.5 套接字缓冲区的上层支持例程	467
12.5 网络设备接口	468
12.5.1 基本结构	468
12.5.2 命名规则	469
12.5.3 设备注册	469
12.5.4 网络设备数据结构	470
12.5.5 支持函数	473
第十三章 Linux 启动系统	477
13.1 初始化流程	477
13.1.1 系统加电或复位	478
13.1.2 BIOS 启动	478
13.1.3 Boot Loader	479
13.1.4 操作系统的初始化	479
13.2 初始化的任务	479
13.2.1 处理器对初始化的影响	479
13.2.2 其他硬件设备对处理器的影响	480
13.3 Linux 的 Boot Loader	480
13.3.1 软盘的结构	480
13.3.2 硬盘的结构	481
13.3.3 Boot Loader	481

13.3.4 LILO	482
13.3.5 LILO 的运行分析	485
13.4 进入操作系统	487
13.4.1 Setup.S	487
13.4.2 Head.S	488
13.5 main.c 中的初始化	491
13.6 建立 init 进程	495
13.6.1 init 进程的建立	495
13.6.2 启动所需的 Shell 脚本文件	497
附录 A Linux 内核 API	501
附录 B 在线文档	529
参考文献	531