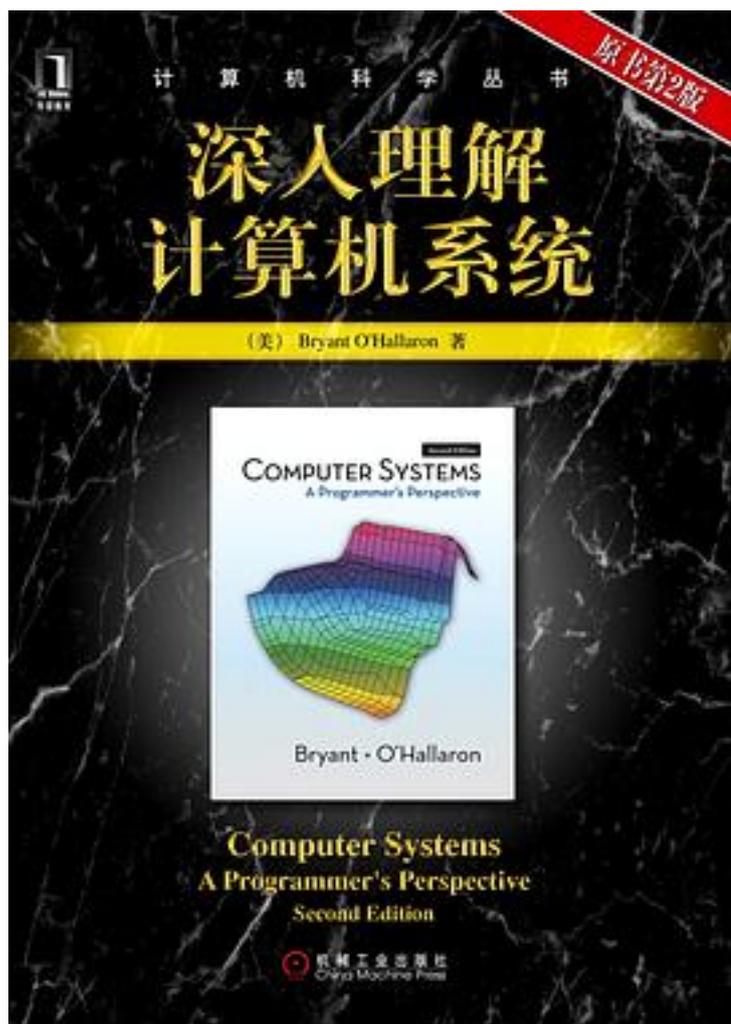


深入理解计算机系统（原书第2版）



[深入理解计算机系统（原书第2版） 下载链接1](#)

著者: (美) Randal E. Bryant

出版者: 机械工业出版社

出版时间: 2011-1-1

装帧: 平装

isbn: 9787111321330

本书从程序员的视角详细阐述计算机系统的本质概念，并展示这些概念如何实实在在地

影响应用程序的正确性、性能和实用性。全书共12章，主要内容包括信息的表示和处理、程序的机器级表示、处理器体系结构、优化程序性能、存储器层次结构、链接、异常控制流、虚拟存储器、系统级I/O、网络编程、并发编程等。书中提供大量的例子和练习，并给出部分答案，有助于读者加深对正文所述概念和知识的理解。

本书的最大优点是程序员描述计算机系统的实现细节，帮助其在大脑中构造一个层次型的计算机系统，从最底层的数据在内存中的表示到流水线指令的构成，到虚拟存储器，到编译系统，到动态加载库，到最后的用户态应用。通过掌握程序是如何映射到系统上，以及程序是如何执行的，读者能够更好地理解程序的行为为什么是这样的，以及效率低下是如何造成的。

本书适合那些想要写出更快、更可靠程序的程序员阅读，也适合作为高等院校计算机及相关专业本科生、研究生的教材。

作者简介:

Randal E. Bryant 1973年于密歇根大学 (University of Michigan) 获得学士学位，随即就读于麻省理工学院的研究生院，并在1981年获计算机博士学位。他在加州理工学院 (California Institute of Technology) 做了三年助教，从1984年至今一直是卡内基-梅隆大学的教师。他现在是计算机科学的大学教授和计算机科学学院的院长。他同时还受邀于电子和计算机工程系。

他从事本科生和研究生计算机系统方面课程的教学超过30年。在讲授计算机体系结构课程多年后，他开始把关注点从如何设计计算机转移到程序员如何在更好的了解系统的情况下编写出更有效和更可靠的程序。他和O' Hallaron教授一起在卡内基梅隆大学开设了15-213 “计算机系统导论”课程，那便是此书的基础。他还教授一些有关算法、编程、计算机网络和VLSI (超大规模集成电路) 设计方面的课程。

Bryant教授的主要研究内容是设计软件工具来帮助软件和硬件设计者验证其系统正确性。其中，包括几种类型的模拟器，以及用数学方法来证明设计正确性的形式化验证工具。他发表了150多篇技术论文。包括Intel、FreeScale、IBM和Fujitsu在内的主要计算机制造商都使用着他的研究成果。他还因他的研究获得过数项大奖。其中包括Semiconductor Research Corporation颁发的两个发明荣誉奖和一个技术成就奖，ACM颁发的Kanellakis理论与实践奖，还有IEEE授予的W.R.G. Baker奖、Emmanuel Piore奖和Phil Kaufman奖。他还是ACM院士、IEEE院士和美国国家工程院院士。

David R. O' Hallaron

现为Intel匹兹堡实验室主任，卡内基-梅隆大学电子和计算机工程系副教授。在弗吉尼亚大学获得计算机科学的博士学位。

他教授本科生和研究生的计算机系统方面的课程，例如计算机体系结构、计算机系统导论、并行处理器设计和Internet服务。他和Bryant教授一起开设了“计算机系统导论”课程，那便是此书的基础。2004年他获得了CMU计算机学院颁发的Herbert Simon杰出教学奖，这个奖项的获得者是基于学生的投票产生的。

O' Hallaron教授从事计算机系统领域的研究，主要兴趣在于科学计算、数据密集型计算和虚拟化方面的软件系统。其中最著名的是Quake项目，一群计算机科学家、土木工程师和地震学家致力于提高对强烈地震中大地运动的预测能力。2003年，他同Quake项目中其他成员一起获得了高性能计算领域中的最高国际奖项—Gordon Bell奖。

目录: 出版者的话

译者序

前言

第1章 计算机系统漫游1

1.1 信息就是位+上下文1

1.2 程序被其他程序翻译成不同的格式3

1.3 了解编译系统如何工作是大有益处的4

1.4 处理器读并解释存储在存储器中的指令5

1.4.1 系统的硬件组成5

1.4.2 运行hello程序7

1.5 高速缓存至关重要7

1.6 存储设备形成层次结构9

1.7 操作系统管理硬件10

1.7.1 进程11

1.7.2 线程12

1.7.3 虚拟存储器12

1.7.4 文件13

1.8 系统之间利用网络通信13

1.9 重要主题15

1.9.1 并发和并行15

1.9.2 计算机系统中抽象的重要性17

1.10 小结17

参考文献说明18

第一部分 程序结构和执行

第2章 信息的表示和处理20

2.1 信息存储22

2.1.1 十六进制表示法22

2.1.2 字25

2.1.3 数据大小25

2.1.4 寻址和字节顺序26

2.1.5 表示字符串31

2.1.6 表示代码31

2.1.7 布尔代数简介32

2.1.8 C语言中的位级运算34

2.1.9 C语言中的逻辑运算36

2.1.10 C语言中的移位运算36

2.2 整数表示38

2.2.1 整型数据类型38

2.2.2 无符号数的编码39

2.2.3 补码编码40

2.2.4 有符号数和无符号数之间的转换44

2.2.5 C语言中的有符号数与无符号数47

2.2.6 扩展一个数字的位表示49

2.2.7 截断数字51

2.2.8 关于有符号数与无符号数的建议52

2.3 整数运算54

2.3.1 无符号加法54

2.3.2 补码加法57

2.3.3 补码的非59

2.3.4 无符号乘法60

2.3.5 补码乘法60

2.3.6 乘以常数63

2.3.7 除以2的幂64

2.3.8 关于整数运算的最后思考67

| | |
|------------------|-----|
| 2.4 浮点数 | 67 |
| 2.4.1 二进制小数 | 68 |
| 2.4.2 IEEE浮点表示 | 70 |
| 2.4.3 数字示例 | 71 |
| 2.4.4 舍入 | 74 |
| 2.4.5 浮点运算 | 76 |
| 2.4.6 C语言中的浮点数 | 77 |
| 2.5 小结 | 79 |
| 参考文献说明 | 80 |
| 家庭作业 | 80 |
| 练习题答案 | 90 |
| 第3章 程序的机器级表示 | 102 |
| 3.1 历史观点 | 103 |
| 3.2 程序编码 | 105 |
| 3.2.1 机器级代码 | 106 |
| 3.2.2 代码示例 | 107 |
| 3.2.3 关于格式的注解 | 109 |
| 3.3 数据格式 | 111 |
| 3.4 访问信息 | 112 |
| 3.4.1 操作数指示符 | 112 |
| 3.4.2 数据传送指令 | 114 |
| 3.4.3 数据传送示例 | 116 |
| 3.5 算术和逻辑操作 | 118 |
| 3.5.1 加载有效地址 | 118 |
| 3.5.2 一元操作和二元操作 | 119 |
| 3.5.3 移位操作 | 120 |
| 3.5.4 讨论 | 120 |
| 3.5.5 特殊的算术操作 | 122 |
| 3.6 控制 | 123 |
| 3.6.1 条件码 | 124 |
| 3.6.2 访问条件码 | 125 |
| 3.6.3 跳转指令及其编码 | 127 |
| 3.6.4 翻译条件分支 | 129 |
| 3.6.5 循环 | 132 |
| 3.6.6 条件传送指令 | 139 |
| 3.6.7 switch语句 | 144 |
| 3.7 过程 | 149 |
| 3.7.1 栈帧结构 | 149 |
| 3.7.2 转移控制 | 150 |
| 3.7.3 寄存器使用惯例 | 151 |
| 3.7.4 过程示例 | 152 |
| 3.7.5 递归过程 | 156 |
| 3.8 数组分配和访问 | 158 |
| 3.8.1 基本原则 | 158 |
| 3.8.2 指针运算 | 159 |
| 3.8.3 嵌套的数组 | 159 |
| 3.8.4 定长数组 | 161 |
| 3.8.5 变长数组 | 163 |
| 3.9 异质的数据结构 | 164 |
| 3.9.1 结构 | 164 |
| 3.9.2 联合 | 167 |
| 3.9.3 数据对齐 | 170 |
| 3.10 综合：理解指针 | 172 |
| 3.11 应用：使用GDB调试器 | 174 |

- 3.12 存储器的越界引用和缓冲区溢出175
- 3.13 x86-64: 将IA32扩展到64位183
 - 3.13.1 x86-64的历史和动因184
 - 3.13.2 x86-64简介185
 - 3.13.3 访问信息187
 - 3.13.4 控制192
 - 3.13.5 数据结构200
 - 3.13.6 关于x86-64的总结性评论200
- 3.14 浮点程序的机器级表示201
- 3.15 小结201
- 参考文献说明202
- 家庭作业202
- 练习题答案212
- 第4章 处理器体系结构230
 - 4.1 Y86指令集体系结构231
 - 4.1.1 程序员可见的状态231
 - 4.1.2 Y86指令232
 - 4.1.3 指令编码233
 - 4.1.4 Y86异常237
 - 4.1.5 Y86程序237
 - 4.1.6 一些Y86指令的详情241
 - 4.2 逻辑设计和硬件控制语言HCL242
 - 4.2.1 逻辑门243
 - 4.2.2 组合电路和HCL布尔表达式243
 - 4.2.3 字级的组合电路和HCL整数表达式245
 - 4.2.4 集合关系248
 - 4.2.5 存储器和时钟248
 - 4.3 Y86的顺序实现250
 - 4.3.1 将处理组织成阶段250
 - 4.3.2 SEQ硬件结构258
 - 4.3.3 SEQ的时序259
 - 4.3.4 SEQ阶段的实现262
 - 4.4 流水线的通用原理267
 - 4.4.1 计算流水线268
 - 4.4.2 流水线操作的详细说明269
 - 4.4.3 流水线的局限性271
 - 4.4.4 带反馈的流水线系统272
 - 4.5 Y86的流水线实现273
 - 4.5.1 SEQ+: 重新安排计算阶段273
 - 4.5.2 插入流水线寄存器276
 - 4.5.3 对信号进行重新排列和标号277
 - 4.5.4 预测下一个PC279
 - 4.5.5 流水线冒险280
 - 4.5.6 用暂停来避免数据冒险283
 - 4.5.7 用转发来避免数据冒险285
 - 4.5.8 加载/使用数据冒险288
 - 4.5.9 异常处理289
 - 4.5.10 PIPE各阶段的实现291
 - 4.5.11 流水线控制逻辑297
 - 4.5.12 性能分析305
 - 4.5.13 未完成的工作306
 - 4.6 小结308
 - 参考文献说明309
 - 家庭作业309

练习题答案314

第5章 优化程序性能324

5.1 优化编译器的能力和局限性325

5.2 表示程序性能328

5.3 程序示例330

5.4 消除循环的低效率332

5.5 减少过程调用336

5.6 消除不必要的存储器引用336

5.7 理解现代处理器340

5.7.1 整体操作340

5.7.2 功能单元的性能343

5.7.3 处理器操作的抽象模型344

5.8 循环展开348

5.9 提高并行性351

5.9.1 多个累积变量351

5.9.2 重新结合变换354

5.10 优化合并代码的结果小结358

5.11 一些限制因素359

5.11.1 寄存器溢出359

5.11.2 分支预测和预测错误处罚360

5.12 理解存储器性能363

5.12.1 加载的性能363

5.12.2 存储的性能364

5.13 应用：性能提高技术369

5.14 确认和消除性能瓶颈369

5.14.1 程序剖析370

5.14.2 使用剖析程序来指导优化371

5.14.3 Amdahl定律374

5.15 小结375

参考文献说明375

家庭作业376

练习题答案378

第6章 存储器层次结构382

6.1 存储技术382

6.1.1 随机访问存储器383

6.1.2 磁盘存储389

6.1.3 固态硬盘398

6.1.4 存储技术趋势399

6.2 局部性401

6.2.1 对程序数据引用的局部性402

6.2.2 取指令的局部性403

6.2.3 局部性小结403

6.3 存储器层次结构405

6.3.1 存储器层次结构中的缓存406

6.3.2 存储器层次结构概念小结408

6.4 高速缓存存储器408

6.4.1 通用的高速缓存存储器结构409

6.4.2 直接映射高速缓存410

6.4.3 组相联高速缓存416

6.4.4 全相联高速缓存418

6.4.5 有关写的问题420

6.4.6 一个真实的高速缓存层次结构的解剖421

6.4.7 高速缓存参数的性能影响422

6.5 编写高速缓存友好的代码423

- 6.6 综合：高速缓存对程序性能的影响426
 - 6.6.1 存储器山426
 - 6.6.2 重新排列循环以提高空间局部性430
 - 6.6.3 在程序中利用局部性433
- 6.7 小结433
- 参考文献说明434
- 家庭作业434
- 练习题答案442
- 第二部分 在系统上运行程序
- 第7章 链接448
 - 7.1 编译器驱动程序449
 - 7.2 静态链接450
 - 7.3 目标文件450
 - 7.4 可重定位目标文件451
 - 7.5 符号和符号表452
 - 7.6 符号解析454
 - 7.6.1 链接器如何解析多重定义的全局符号455
 - 7.6.2 与静态库链接457
 - 7.6.3 链接器如何使用静态库来解析引用460
 - 7.7 重定位461
 - 7.7.1 重定位条目461
 - 7.7.2 重定位符号引用462
 - 7.8 可执行目标文件465
 - 7.9 加载可执行目标文件466
 - 7.10 动态链接共享库467
 - 7.11 从应用程序中加载和链接共享库468
 - 7.12 与位置无关的代码 (PIC) 471
 - 7.13 处理目标文件的工具473
 - 7.14 小结473
- 参考文献说明474
- 家庭作业474
- 练习题答案479
- 第8章 异常控制流480
 - 8.1 异常481
 - 8.1.1 异常处理481
 - 8.1.2 异常类别482
 - 8.1.3 Linux/IA32系统中的异常484
 - 8.2 进程487
 - 8.2.1 逻辑控制流487
 - 8.2.2 并发流487
 - 8.2.3 私有地址空间488
 - 8.2.4 用户模式和内核模式488
 - 8.2.5 上下文切换489
 - 8.3 系统调用错误处理491
 - 8.4 进程控制492
 - 8.4.1 获取进程ID492
 - 8.4.2 创建和终止进程492
 - 8.4.3 回收子进程495
 - 8.4.4 让进程休眠499
 - 8.4.5 加载并运行程序500
 - 8.4.6 利用fork和execve运行程序502
 - 8.5 信号504
 - 8.5.1 信号术语505
 - 8.5.2 发送信号506

- 8.5.3 接收信号509
- 8.5.4 信号处理问题511
- 8.5.5 可移植的信号处理516
- 8.5.6 显式地阻塞和取消阻塞信号517
- 8.5.7 同步流以避免讨厌的并发错误517
- 8.6 非本地跳转521
- 8.7 操作进程的工具524
- 8.8 小结524
- 参考文献说明525
- 家庭作业525
- 练习题答案530
- 第9章 虚拟存储器534
- 9.1 物理和虚拟寻址535
- 9.2 地址空间535
- 9.3 虚拟存储器作为缓存的工具536
- 9.3.1 DRAM缓存的组织结构537
- 9.3.2 页表537
- 9.3.3 页命中538
- 9.3.4 缺页538
- 9.3.5 分配页面539
- 9.3.6 又是局部性救了我们539
- 9.4 虚拟存储器作为存储器管理的工具540
- 9.5 虚拟存储器作为存储器保护的工具541
- 9.6 地址翻译542
- 9.6.1 结合高速缓存和虚拟存储器544
- 9.6.2 利用TLB加速地址翻译545
- 9.6.3 多级页表546
- 9.6.4 综合：端到端的地址翻译547
- 9.7 案例研究：Intel Core i7/Linux存储器系统550
- 9.7.1 Core i7地址翻译551
- 9.7.2 Linux虚拟存储器系统554
- 9.8 存储器映射556
- 9.8.1 再看共享对象557
- 9.8.2 再看fork函数558
- 9.8.3 再看execve函数559
- 9.8.4 使用mmap函数的用户级存储器映射559
- 9.9 动态存储器分配561
- 9.9.1 malloc和free函数561
- 9.9.2 为什么要使用动态存储器分配563
- 9.9.3 分配器的要求和目标564
- 9.9.4 碎片565
- 9.9.5 实现问题565
- 9.9.6 隐式空闲链表565
- 9.9.7 放置已分配的块567
- 9.9.8 分割空闲块567
- 9.9.9 获取额外的堆存储器567
- 9.9.10 合并空闲块568
- 9.9.11 带边界标记的合并568
- 9.9.12 综合：实现一个简单的分配器570
- 9.9.13 显式空闲链表576
- 9.9.14 分离的空闲链表576
- 9.10 垃圾收集578
- 9.10.1 垃圾收集器的基本知识579
- 9.10.2 Mark&Sweep垃圾收集器580

- 9.10.3 C程序的保守Mark&Sweep580
- 9.11 C程序中常见的与存储器有关的错误581
 - 9.11.1 间接引用坏指针582
 - 9.11.2 读未初始化的存储器582
 - 9.11.3 允许栈缓冲区溢出582
 - 9.11.4 假设指针和它们指向的对象是相同大小的583
 - 9.11.5 造成错位错误583
 - 9.11.6 引用指针，而不是它所指向的对象583
 - 9.11.7 误解指针运算584
 - 9.11.8 引用不存在的变量584
 - 9.11.9 引用空闲堆块中的数据584
 - 9.11.10 引起存储器泄漏585
- 9.12 小结585
- 参考文献说明586
- 家庭作业586
- 练习题答案589
- 第三部分 程序间的交互和通信
- 第10章 系统级I/O596
 - 10.1 Unix I/O596
 - 10.2 打开和关闭文件597
 - 10.3 读和写文件598
 - 10.4 用RIO包健壮地读写599
 - 10.4.1 RIO的无缓冲的输入输出函数600
 - 10.4.2 RIO的带缓冲的输入函数600
 - 10.5 读取文件元数据604
 - 10.6 共享文件606
 - 10.7 I/O重定向608
 - 10.8 标准I/O609
 - 10.9 综合：我该使用哪些I/O函数610
 - 10.10 小结611
 - 参考文献说明612
 - 家庭作业612
 - 练习题答案612
- 第11章 网络编程614
 - 11.1 客户端-服务器编程模型614
 - 11.2 网络615
 - 11.3 全球IP因特网618
 - 11.3.1 IP地址619
 - 11.3.2 因特网域名620
 - 11.3.3 因特网连接623
 - 11.4 套接字接口625
 - 11.4.1 套接字地址结构625
 - 11.4.2 socket函数626
 - 11.4.3 connect函数626
 - 11.4.4 open_clientfd函数627
 - 11.4.5 bind函数628
 - 11.4.6 listen函数628
 - 11.4.7 open_listenfd函数628
 - 11.4.8 accept函数629
 - 11.4.9 echo客户端和服务器的示例630
 - 11.5 Web服务器633
 - 11.5.1 Web基础633
 - 11.5.2 Web内容633
 - 11.5.3 HTTP事务634

11.5.4 服务动态内容636
11.6 综合：TINY Web服务器639
11.7 小结645
参考文献说明645
家庭作业646
练习题答案646
第12章 并发编程648
12.1 基于进程的并发编程649
12.1.1 基于进程的并发服务器649
12.1.2 关于进程的优劣651
12.2 基于I/O多路复用的并发编程651
12.2.1 基于I/O多路复用的并发事件驱动服务器653
12.2.2 I/O多路复用技术的优劣657
12.3 基于线程的并发编程657
12.3.1 线程执行模型657
12.3.2 Posix线程658
12.3.3 创建线程659
12.3.4 终止线程659
12.3.5 回收已终止线程的资源660
12.3.6 分离线程660
12.3.7 初始化线程660
12.3.8 一个基于线程的并发服务器661
12.4 多线程程序中的共享变量662
12.4.1 线程存储器模型663
12.4.2 将变量映射到存储器663
12.4.3 共享变量664
12.5 用信号量同步线程664
12.5.1 进度图667
12.5.2 信号量668
12.5.3 使用信号量来实现互斥669
12.5.4 利用信号量来调度共享资源670
12.5.5 综合：基于预线程化的并发服务器674
12.6 使用线程提高并行性676
12.7 其他并发问题680
12.7.1 线程安全680
12.7.2 可重入性682
12.7.3 在线程化的程序中使用已存在的库函数682
12.7.4 竞争683
12.7.5 死锁685
12.8 小结687
参考文献说明687
家庭作业688
练习题答案691
附录A 错误处理694
A.1 Unix系统中的错误处理694
A.2 错误处理包装函数696
参考文献698
• • • • • [\(收起\)](#)

[深入理解计算机系统（原书第2版）_下载链接1](#)

标签

计算机系统

计算机

计算机科学

操作系统

编程

体系结构

程序员

评论

值得翻来覆去读100遍的好书，为啥要读那么多遍呢？因为反复都读不很懂啊。。。

花了一个月时间看完这本大部头，此书貌似是豆瓣评分最高的计算机书籍，经我亲自鉴定，确实是本好书，必须强烈推荐，吐血推荐，五星级推荐！准备回学校以后强烈建议导师将此书列为进实验室必读书籍，通过阅读经典书籍，提升师弟师妹的读书品味，也算是作为师兄，为实验室做了点贡献。

计算机科学的两本圣经之一，一本十分精彩的书。学完会很累，但也能学到很多有用的东西。

CMU计算机系基础课程，计算机科学的两本圣经之一，全书都是从程序员角度出发去

了解计算机底层系统，避免在一些硬件层问题的无谓纠缠，为以后操作系统、编译原理、体系结构奠定了基础。不过和之前课程脱节挺严重的，计概程设注重算法，缺乏对计算机基础问题的讲解，导致课程一开始有些吃力。课程附带的Lab是噩梦之源，平均每
周几十面的英文说明+几百行的代码量+无数次翻墙测试，配合数算实习就可以填满周末了。不谈最后虐翻全系的考试，一个学期下来还是收获颇多，对底层的了解可能不会对算法层面产生很大影响，但对程序性能的优化和解决执行中的系统、硬件问题还是有非常多的帮助。收获最多的有数据储存、汇编和机器码、缓存系统、异常和信号处理、虚存和动态储存器、系统I/O、网络模型、进程与线程、并发编程。

因为太厚了,被我劈为两半.这样就可以随身带着一部分.

ICS课大概是我在FDU上过最硬的课

每年读一遍，每遍都有新收获

体系结构关键的是理解图灵机模型，但本书的困难在于体系结构和操作系统的大量交叉组合，对于初学者不利，初学者可以阅读《计算机组成结构化方法》和《现代操作系统》，但是对于有一定基础的人来说，按图索骥就成为了一种思维的享受。系统本就是软硬件的交互作用后的逻辑：计算机是计算的核心，如果没有计算机，今天的大多数计算机科学只是理论数学的分支。

这本书对计算机各个方面进行了全面而深入的介绍，作者们用非常认真细致的态度，给出异常清晰的讲解。看完后，我觉得：当年我的老师们似乎并不是特别理解他们所讲的东西。

要是早个三四年看过这本书，许多问题和困扰就不会出现了。

走向高帅富第一步

别的教材没见过有把2的补码讲得这么清楚的

Definiter: 「因为那是个复杂、精细、不美的庞然大物」。

其实这本书到现在也没翻过几页……………下个学期的数据库刘攀你敢不敢再让我买一本废书不讲书上的东西!

汇编和体系结构基本没看，早点看完最后一章的话笔试面试应该会表现得更好一些 - =

从C开始理解计算机系统，从计算机系统中理解C语言。要是早看了微机原理分不至于那么低，但是没有一定代码量的积累也看不懂这本书。值得细细笔记的好书，而且不知道为什么，这样书越看越有味道。

12/11 - 13/1 读后功力大增，相见恨晚
能把本质的东西讲的这么易懂，有种读小说的感觉

有点惭愧，花了快一年的时间才读完。读完又有点惋惜，这么好的书怕是再难找到了。作者对阅读材料安排得当，每天花30到40分钟，刚好可以看一小节的内容，顺便做完每节后的习题。汇编的一章，加强了对机器底层的理解，涉及到指令级并行，后续的虚存和异常流一节，对于应用级的并发则很有裨益。

很经典的一本书，虽然有的地方有点忘记了，但非常不错，标记一下。

很明显，这本书我没有看完。前前后后看了7章，程序优化没看，后面的网络也没有看，重点看了前三章以及处理器和内存的章节。

[深入理解计算机系统（原书第2版） 下载链接1](#)

书评

CMU是全美以至全球公认的CS最猛的大学之一，没办法，作为CS的发源地，再加上三位神一样的人先后在此任教：Alan Perlis（CS它祖宗+第一届Turing奖获得者）、Allen Newell（AI缔造者+Turing奖获得者）和Herbert Simon（AI缔造者+Turing奖获得者+Nobel经济学奖获得者，当代的L...

最新课程：2014年6月30日这门课在Coursera开始第二季，地址<https://www.coursera.org/course/hwswinterface>，请大家及时关注
如果你觉得这本书过于厚重担心看不下来的话，不妨跟着coursera的Hardware/Software Interface这门课程去听一听。这本书虽然是这门课的超集，但是其中...

Chapter 1 A Tour of Computer System 一个对计算机系统总体的介绍，简单明了。应试★★★：可能在笔试中会有一些整体上的概念题。修炼★：属于计算机最基本的概念。

上个星期终于把csapp看完了。
我买的是中文版的，因为除了貌似评价不错以外，由于涉及到些自己不了解的底层东西，怕是看英文会云里雾里。现在看来，大概不能算是个优点，但是的确能够加快我的看书速度，否则一星期还真不大可能把这书搞定。
对csapp慕名已久，主要在于据说这本...

第一次在豆瓣上发东西，呵呵～
本学期选了汇编程序设计，实际上就是用这本CSAPP当教材了。作者是CMU计算机系主任，该门课的经典地位应该和MIT的SICP差不多吧（也在这学期选了～）。
最初只是想混学分，结果上了之后才发觉这门课是如此强大。我们老师也将CMU原配的几个Lab作业...

这本书的中译名为“深入理解计算机系统”，有一定的问题。如果直译原书名，应该是类似于“以程序员的视角理解计算机系统”。可能在国内看来，这是讲系统的，用C和汇编语言的，因此很“深入”。事实上，这是一本入门级别的书，这本书其实并不“深入”，它谈论的内容还是相对比较...

CMU和ICS的课号为213，然后他的courseweb在这:

<http://www.cs.cmu.edu/~213/index.html>

里面有CMU往年的CS213的所有exam资料和答案:

<http://www.cs.cmu.edu/~213/exams.html> 还有另外一个资源是:

<http://www.cs.cmu.edu/~213/lectures/> 这里是FTP服务器，提供所有CS213的slide...

如果计算机科学只能带走一本书的话，我选择这本。

这是本很有趣的书，了解计算机底层技术是件极富挑战同趣味的事情，这样的书籍给人带来极大乐趣。个人认为这样的书籍，计科专业的本科学生越早接触越好，它会让你比别人更早拥有一个系统的计算机知识，更能触类旁通。如果想从事...

这本书是很好的书，我认为只要是工作中涉及编程工作的同学都要必须要熟读的书。我之前看过第二版，没看完。后来发现有第三版了，我就从网上找来一本二手的，重新从头认真读，现在读完了前5章，简单说一下。

跟之前的中文版第二版相比，这本第三版的印刷质量和纸张都有很大进步...

这是一本多么伟大的书籍！我希望我曾去过卡耐基梅隆大学并参加这门课程。这本书是卡耐基梅隆大学的教授在讲授计算机系统课程后的几年时间里写的。从程序员的角度看（作为标题来说更为恰当），这本书涵盖了广泛的主题范围，包括操作系统，编译器，计算机系统结构，集级编程，内...

直到今天,大体上看完了这本书,空过了其中四章.因为它们说的是比较以Unix为基础的技术或者是网络技术或者比较深入地讨论了某些细节.它们分别是第八章,异常控制流,第九章,测量程序执行时间,第十二章,网络编程,第十三章,并发编程.花了大约四十天,除了其中有些日子,应该是实实...

注：图片无法显示，请参考：

<http://www.cnblogs.com/remlostime/archive/2011/04/10/2011914.html>

最近在上金博的《计算机原理》。为什么说是最值得上的课，原因有二。

一者，教材是CMU的人写的久负盛名的《Computer Systems:A Programmer's Perspective》（<http://book.do...>

这本书的中文版是从原书第二版翻译的，第二版是2001年出版的。
这本书是基于IA32架构的，而目前大部分的计算机都是基于x86_64的，如果你已经身经百战见得多了可能并不在意这些区别，但是要是初学的话发现自己电脑上跑出来的效果和书上相差比较大还是会很懵逼的。这本书第三...

越来越觉得，这本书的价值远远超过我网购总价 84 元。
翻译、印刷、排版、纸张等等都非常地好，远比第一版好多了。
它使我一点点地明白了以前未知的、理解模糊的重要知识点。这学期刚好在上“计算机系统结构”这门课，用的是张晨曦老师的教材，主任一直夸这本教材是目前国内...

刚读完这本书. 感觉很像是计算机导论
将计算机与硬件相关的入门知识都笼统的介绍了一下 讲解了他们怎么用
如果有数学基础 例如数学物理生物等立刻专业的同学, 看完这本书再看看组成原理
体系结构 编译原理 还有汇编语言和操作系统原理,
计算机底层的基础知识就已经非常足够了 ...

作为一名计算机程序员，如果缺乏对计算机的层次理解，那么其基本素养是值得怀疑的，其思考基础是有欠缺的。
当我们沉浸在这样或者那样的编程教程里，沉浸在大量的语法架构之中的时候，反而常常因为这样或者那样一门复杂的技术而忘记计算机系统的构成其本身。

也是一个偶然的的机会才在别人的书桌上随便翻开看看的。结果一发不可收拾，现在自己花了RMB72购入囊中，列入珍藏的书目中了。正如英文的原名所叙述的，from a programmer's perspective,
顾名思义，就是从程序员的视角来看待一个计算机系统。现有的一些计算机原理书，往往过于偏...

英文名：Computer Systems : A Programmer's Perspective 作者：【美】Randal E.Bryant、David O'Hallaron 序言 第1章 计算机系统漫游
计算机系统是由硬件和系统软件组成的，它们共同协作以运行应用程序。计算机内部的信息被表示为一组组的位，它们依据不同的上下文又有...

[深入理解计算机系统（原书第2版）_下载链接1](#)