

Mc
Graw
Hill Education

学习指导与习题解答

Schaum's Outline of Computer Architecture



计算机体系结构 学习指导与习题解答

Nicholas Carter 著

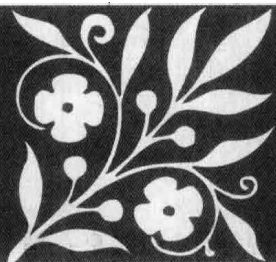
马江涛 等 译

清华大学出版社



学习指导与习题解答

Schaum's Outline of Computer Architecture



计算机体系结构
学习指导与习题解答

清华大学出版社
北京

Nicholas Carter

Schaum's Outline of Computer Architecture

EISBN: 978-7-07-136207-8

Copyright © 2010 The McGraw-Hill Companies, Inc.

All Rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including without limitation photocopying, recording, taping, or any database, information or retrieval system, without the prior written permission of the publisher.

This authorized Chinese translation edition is jointly published by McGraw-Hill Education (Asia) and Tsinghua University Press. This edition is authorized for sale in the People's Republic of China only, excluding Hong Kong, Macao SAR and Taiwan.

Copyright © 2010 by McGraw-Hill Education (Asia), a division of the Singapore Branch of The McGraw-Hill Companies, Inc. and Tsinghua University Press.

版权所有。未经出版人事先书面许可,对本出版物的任何部分不得以任何方式或途径复制或传播,包括但不限于复印、录制、录音,或通过任何数据库、信息或可检索的系统。

本授权中文简体字翻译版由麦格劳-希尔(亚洲)教育出版公司和清华大学出版社合作出版。此版本经授权仅限在中华人民共和国境内(不包括香港特别行政区、澳门特别行政区和中国台湾地区)销售。

版权© 2010 由麦格劳-希尔(亚洲)教育出版公司与清华大学出版社所有。

北京市版权局著作权合同登记号 图字:01-2009-5139号

本书封面贴有 McGraw-Hill 公司防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

计算机体系结构学习指导与习题解答/(美)卡特(Carter,N.)著;马江涛等译. —北京:清华大学出版社,2010.8

书名原文:Schaum's Outline of Computer Architecture

ISBN 978-7-302-22882-0

I. ①计… II. ①卡… ②马… III. ①计算机体系结构—高等学校—教学参考资料
IV. ①TP303

中国版本图书馆 CIP 数据核字(2010)第 099226 号

责任编辑:龙啟铭

责任校对:李建庄

责任印制:王秀菊

出版发行:清华大学出版社

<http://www.tup.com.cn>

社总机:010-62770175

投稿与读者服务:010-62795954,jsjic@tup.tsinghua.edu.cn

质量反馈:010-62772015,zhiliang@tup.tsinghua.edu.cn

地址:北京清华大学学研大厦 A 座

邮编:100084

邮购:010-62786544

印装者:北京市清华园胶印厂

经销:全国新华书店

开本:185×230 印张:16.25

字数:352千字

版次:2010年8月第1版

印次:2010年8月第1次印刷

印数:1~3000

定 价:33.00元

产品编号:034574-01

译者序

计算机系统结构是计算机及相关学科的专业基础课程，它强调从总体结构、系统分析这一角度来研究计算机系统，对培养学生自上而下系统分析和解决问题的能力以及培养学生的抽象思维能力有着非常重要的作用。计算机体系结构的经典定义是在1964年由C.M. Amdahl提出的，具体描述为“计算机体系结构是程序员所看到的计算机的属性，即概念性结构与功能特性”。

本书系统地介绍计算机体系结构的基础知识，内容实用而丰富，主要内容有：

第1章讨论影响计算机性能的基本技术和计算机性能的量度方法，为其他章节的学习奠定基础。

第2章说明计算机系统最常用的数据表示方法以及算术运算方法。

第3章介绍构成计算机的基本组成，包括处理器、存储器和I/O系统等，说明计算机体系结构的实质。本章还简要描述计算机系统如何在内部表示程序，操作系统如何调度程序，以及如何控制组成计算机的各种物理设备。

第4章讲解两类处理器的编程模型：基于栈的体系结构和通用寄存器体系结构。

第5章介绍处理器设计，详细讲解前几章涉及的抽象概念。

第6章的主要内容是改进处理器性能的流水线方法。利用流水线方法，处理器可以重叠执行若干条指令，在相同时间段内就可以执行更多的指令。

第7章讨论通过同时执行独立指令实现指令级并行性的各种技术。

第8章讨论存储器系统。

第9章介绍Cache，它是计算机中位置接近于处理器的小型快速存储器。

第10章介绍虚拟内存技术，可将诸如硬盘这样的磁介质存储器作为系统内存，为在相同系统上运行的不同程序之间提供保护，使一个程序不能修改另一个程序的数据。

第11章讨论输入输出（I/O）设备如何与处理器和内存系统进行交互。

第12章介绍多处理器系统，对有关计算机体系结构和组织的讨论进行总结。

本书的最大特点是通过大量丰富的示例和习题，并给出详尽的答案解析，让读者全面透彻地理解概念，掌握每个知识点。

本书主要由马江涛翻译，参加翻译工作的人员还有李志云、李晓春、陈安华、侯佳宜、许伟、戴文雅、于樊鹏、刘朋、王嘉佳、邓卫、邓凡平、李波、程云建、许晓哲、朱珂、韦笑、孙宏、李腾、陈磊、魏宇、周京平、徐冬、冯哲、李绯、李强、赵东辉、王雷、龚亚萍等人。

译者

前言

就计算机体系结构而言，最有意思的一个方面是这个领域变化的速度，可以说是日新月异，为从事这个领域的人们提供了大量机会。然而，对于教授计算机体系结构和组织的教师来说，技术发展迅猛也是巨大的挑战。与其他许多领域不同的是，每个学期这门课程都要有所变化才能跟上新技术的发展，而且不会让学生负担过重。同样，编写相关的教材也困难重重，因为作者必须要在新旧技术之间进行权衡。

本书考虑到以前接触过这个领域的读者的需求，对主题做了精心的选择。第 1~5 章讲解计算机组织的大量基本概念，包括性能量度方法、计算机表示数字数据和程序的方式、计算机的不同编程模型，以及基本的处理器设计。第 6 章和第 7 章介绍流水线以及指令级并行性，对现代处理器性能而言，这两种技术非常重要。第 8~10 章介绍内存系统设计，包括内存层次结构、Cache 以及虚拟内存。第 11 章讲解 I/O 系统。第 12 章介绍多处理器系统，也就是计算机中有多个处理器，以改进性能。

希望本书对读者的学习有帮助。我尽可能清晰地解释每个主题，同时避免陷入对细枝末节的纠缠。在本书如此短的篇幅，讲述计算机体系结构和组织的内容无异于一场挑战。读者如果对主题选择、课后习题以及其他方面有建议和意见，望不吝指教。

最后，感谢为出版本书做出贡献的所有人员，包括我的父母、朋友、伊利诺伊大学的同事以及我的所有老师。特别感谢 McGraw-Hill 出版公司的工作人员，感谢他们的鼓励，感谢他们对我拖延时间的宽容。

Nicholas P. Carter

目 录

第 1 章 概述	1
1.1 本书学习目标.....	1
1.2 读者知识背景.....	1
1.3 本书主要内容.....	1
1.4 本章目标.....	1
1.5 技术趋势.....	2
1.6 性能量度.....	2
1.6.1 MIPS.....	3
1.6.2 CPI/IPC.....	3
1.6.3 基准测试套件.....	4
1.6.4 几何平均数与算术平均数.....	4
1.7 加速比.....	5
1.8 Amdahl 定律.....	5
1.9 本章小结.....	6
本章习题与解析.....	6
第 2 章 数据表示与计算机算术运算	14
2.1 本章学习目标.....	14
2.2 从电子信号到位.....	14
2.3 正整数的二进制表示.....	15
2.4 正整数的算术运算.....	16
2.4.1 加法和减法.....	16
2.4.2 乘法.....	18
2.4.3 除法.....	18
2.4.4 上溢或下溢.....	19
2.5 负整数.....	19
2.5.1 原码表示法.....	19
2.5.2 二进制补码表示法.....	20

2.5.3 符号扩展	22
2.6 浮点数	22
2.6.1 非数字与非规格化数	25
2.6.2 浮点数的算术运算	26
2.7 本章小结	28
本章习题与解析	29
第3章 计算机组织	37
3.1 本章学习目标	37
3.2 本章导言	37
3.3 程序	38
程序开发工具	39
3.4 操作系统	41
3.4.1 多道程序设计	41
3.4.2 保护	42
3.4.3 特权模式	42
3.5 计算机组织	43
3.5.1 处理器	43
3.5.2 内存系统	44
3.5.3 I/O 子系统	45
3.6 本章小结	46
本章习题与解析	46
第4章 编程模型	51
4.1 本章目标	51
4.2 本章导言	51
4.3 指令类型	52
4.3.1 算术操作	52
4.3.2 内存操作	54
4.3.3 比较操作	55
4.3.4 控制操作	56
4.4 基于栈的体系结构	57
4.4.1 栈	57
4.4.2 实现栈	58

4.4.3	基于栈的体系结构的指令	60
4.4.4	基于栈的指令集	60
4.4.5	基于栈的体系结构中的程序	62
4.5	通用寄存器体系结构	63
4.5.1	通用寄存器体系结构中的指令	64
4.5.2	GPR 指令集	65
4.5.3	通用寄存器体系结构的程序	66
4.6	比较基于栈的体系结构与通用寄存器体系结构	67
4.7	利用栈实现过程调用	68
	调用约定	69
4.8	本章小结	69
	本章习题与解析	70
第 5 章	处理器设计	77
5.1	本章目标	77
5.2	本章导言	77
5.3	指令集体系结构	78
5.3.1	比较 RISC 与 CISC	78
5.3.2	寻址方式	80
5.3.3	多媒体向量指令	82
5.3.4	定长指令编码与变长指令编码	83
5.4	处理器微体系结构	84
5.4.1	执行单元	85
5.4.2	微程序设计	86
5.4.3	寄存器文件设计	86
5.5	本章小结	87
	本章习题与解析	88
第 6 章	流水线	94
6.1	本章目标	94
6.2	本章导言	94
6.3	流水线	95
6.3.1	流水线处理器的周期时间	96
6.3.2	流水线延时	98

6.4 指令相关及其对吞吐率的影响	98
6.4.1 转移	101
6.4.2 结构相关	102
6.4.3 记分板	103
6.5 预测流水线处理器的执行时间	103
6.6 结果转发(旁路)	106
6.7 本章小结	109
本章习题与解析	110
第7章 指令级并行性	119
7.1 本章目标	119
7.2 本章导言	119
7.3 什么是指令级并行性	121
7.4 指令级并行性的局限性	121
7.5 超标量处理器	123
7.6 顺序执行与乱序执行	123
7.6.1 顺序执行处理器的执行时间预测	123
7.6.2 乱序执行处理器的执行时间预测	124
7.6.3 乱序执行处理器的实现问题	125
7.7 寄存器重命名	126
7.8 超长指令字处理器	128
超长指令字的利弊	130
7.9 指令级并行性的编译技术	131
7.9.1 循环展开	131
7.9.2 软件流水线方法	133
7.10 本章小结	134
本章习题与解析	135
第8章 内存系统	146
8.1 本章目标	146
8.2 本章导言	146
8.3 延时、吞吐率和带宽	146
流水线、并行性和预充电	147

8.4	内存层次结构	149
8.4.1	层次级别	150
8.4.2	术语	150
8.4.3	平均访问时间	151
8.5	内存技术	152
8.5.1	内存芯片组织	152
8.5.2	SRAM	153
8.5.3	DRAM	154
8.5.4	DRAM 刷新	154
8.5.5	DRAM 访问时序	155
8.5.6	页面模式以及更新的 DRAM	156
8.6	本章小结	157
	本章习题与解析	158
第 9 章	Cache	165
9.1	本章目标	165
9.2	本章导言	165
9.3	数据 Cache、指令 Cache 和统一 Cache	166
9.4	描述 Cache	167
9.5	容量	167
9.6	行长	167
9.7	相联度	168
9.7.1	全相联 Cache	169
9.7.2	直接映射 Cache	169
9.7.3	组相联 Cache	171
9.8	替换策略	172
9.9	比较写回式 Cache 与直写式 Cache	173
9.10	Cache 实现	175
9.11	标记阵列	175
9.12	命中/缺失逻辑	176
9.13	数据阵列	177
9.14	Cache 缺失访问的分类	178
9.15	多级 Cache	179

9.16	本章小结	180
	本章习题与解析	180
第 10 章	虚拟内存	190
10.1	本章目标	190
10.2	本章导言	190
10.3	地址转换	191
10.4	请求调页与页面交换	193
10.5	页表	193
	10.5.1 多级页表	194
	10.5.2 反向页表	196
10.6	TLB	197
	10.6.1 比较 TLB 缺失与缺页	198
	10.6.2 TLB 组织	198
	10.6.3 超页 (页块)	199
10.7	保护	199
10.8	Cache 与虚拟内存	201
10.9	本章小结	202
	本章习题与解析	203
第 11 章	输入输出	208
11.1	本章目标	208
11.2	本章导言	208
11.3	I/O 总线	209
	访问 I/O 总线	210
11.4	中断	210
	11.4.1 实现中断	211
	11.4.2 中断优先级	212
	11.4.3 比较轮询与中断	212
11.5	内存映射 I/O	213
11.6	DMA	214
11.7	输入输出设备	215
11.8	磁盘系统	216
	11.8.1 硬盘组织结构	217

11.8.2 请求调度	218
11.9 本章小结	220
本章习题与解析	220
第 12 章 多处理器	227
12.1 本章目标	227
12.2 本章导言	227
12.3 加速比与性能	227
12.3.1 加速比的局限性	228
12.3.2 超线性加速比	229
12.4 多处理器系统	230
12.5 消息传递系统	232
12.6 共享内存系统	232
12.6.1 内存一致性模型	233
12.6.2 严格一致性	233
12.6.3 Cache 一致性	234
12.6.4 MESI 协议	235
12.6.5 基于总线的共享内存系统	237
12.6.6 同步	237
12.7 比较消息传递与共享内存	238
12.8 本章小结	239
本章习题与解析	239

第 1 章 概 述

1.1 本书学习目标

本书可作为大学高年级和研究生低年级学生学习计算机体系结构的参考书。读者对象是学习本课程的老师和学生，书中提供了详细的解释、与实践相关问题和大量实例，可帮助读者深入理解相关知识点，为完成作业做充分准备。

1.2 读者知识背景

本书假设读者的知识背景：电子工程或计算机科学程序设计专业的大学二三年级的学生，尚未学过有关计算机组织或者计算机体系结构方面的课程。另外，要熟悉各种计算机操作和术语，能在一定程度上熟练使用高级程序设计语言进行编程。

1.3 本书主要内容

为了增强其适用性，与一学期的计算机体系结构课程相比，本书介绍的内容更为广泛。本书的补充材料对复习内容很有用，也可将其作为更高级主题的入门读物。本书首先讨论数据表示和计算机算术运算。随后，介绍计算机组织和编程模型。第 5~7 章，用三章篇幅讨论处理器设计，包括流水线和指令级并行。随后三章内容与内存系统相关，包括虚拟内存和 Cache。最后两章讨论输入输出系统，并介绍多处理器。

1.4 本章目标

本章讨论影响计算机性能的基本技术和计算机性能的量度方法，为随后其他章节的学习作准备。学完本章内容以及完成习题以后，应该能做到：

- (1) 理解和讨论有关晶体管密度、电路性能以及计算机系统整体性能等方面改进速度的历史情况。
- (2) 理解衡量计算机性能的常用方法。
- (3) 计算计算机系统组成部分发生的变化对系统整体性能的影响。

1.5 技术趋势

20 世纪 80 年代初期以来, 计算机性能的改进一直与集成电路性能的提高有关。集成电路用来实现微处理器、内存芯片以及其他计算机部件。随着时间的推移, 集成电路在以下方面获得了改进: 密度(固定面积的硅芯片上可放多少晶体管和电线)、速度(基本逻辑门和存储设备的运算速度)和面积(可制造的最大集成电路的物理尺寸)。

在过去的 20 年中, 推动计算机性能获得极大增长的原因是芯片速度和密度改进呈几何速度增长, 而不是线性速度。这意味着性能的年增长速度是相对恒定的比率, 而不是绝对常量。硅芯片上晶体管数的年均增长率约为 50%, 晶体管的速度增长情况是基本逻辑门(如与门、或门等)延时的年均下降比例是 13%。计算机性能以几何速度而非线性速度增长, 这就是摩尔定律 (Moore's Law)。

示例 自 20 世纪 70 年代后期以来, 动态 RAM (DRAM) 芯片存储的数据量每 3 年增长 4 倍, 年均增长率是 60%。

从 20 世纪 70 年代后期到 20 世纪 80 年代后期, 微处理器性能主要被制造技术的改进所驱动, 其性能改进的年均增长率是 35%。此后, 尽管半导体制造技术的增长率相对不变, 但微处理器性能改进的年均增长率实际上更高了, 超过了 50%。性能提高的主因是计算机体系结构和组织方面的改进, 这样就使得计算机体系结构能够充分利用集成电路密度增长所带来的好处为微处理器和内存系统增加新特性, 与提高潜在晶体管的速度相比, 获得更高的性能改进。

1.6 性能量度

前面介绍过, 计算机性能随时间推移而不断改进, 但没有给出性能的定义。这是因为在计算机系统中性能 (performance) 这个术语含义模糊。通常, 性能用于描述某系统执行一个或多个程序的速度。系统执行程序所用的时间越短, 性能就越高。

计算机性能的最佳指标是用户执行程序的执行时间。但是, 在决定购买哪台计算机或做出决策之前, 要对特定计算机系统运行的所有程序进行测试, 通常是不切实际的。计算机体系结构设计人员提出了描述计算机性能的许多指标, 本章讨论其中的一部分。针对量度单个计算机子系统的性能, 计算机体系结构设计人员设计了许多指标, 它们将在涉及相关子系统的章节中予以讨论。

记住, 除了性能指标外, 还有许多因素也会影响设计或购买决策。易于编程就是一个很重要的因素, 因为开发特定程序所需要的时间和费用, 也许要比程序开发出来后在执行

时间上带来的差异优势更加重要。另一个重要因素是兼容性。多数程序以二进制映像形式销售,只能运行于特定系列的处理器。当想要的程序不能在某系统上运行时,无论系统执行其他程序的速度有多快,都毫无意义。

1.6.1 MIPS

计算机性能的早期指标是某计算机执行指令的速度,计算方法是程序执行的指令数除以运行程序所需时间,通常表示为每秒百万条指令数(Millions of Instructions Per Second, MIPS)。MIPS 已经不再作为一种性能指标,主要原因是它不能解释不同系统在执行给定程序时需要用不同的指令数。由于计算机的 MIPS 指标不能说明执行给定任务时需要多少条指令,因此它要比不同计算机系统性能的其他指标的用处更小。

1.6.2 CPI/IPC

衡量计算机性能的另一个指标是执行每个指令所要的时钟周期数,即每指令周期数(Cycles Per Instruction, CPI)。计算给定系统程序 CPI 的方法是用执行程序所需要的时钟周期数除以程序运行过程执行的指令数。对于每个周期能执行多条指令的系统而言,通常使用 IPC (Instructions Executed Per Cycle, 每周期执行指令) 指标,而不是 CPI。IPC 的计算方法是用程序运行过程中执行的指令数除以执行程序所需要的时钟周期数,即 CPI 的倒数。这两个指标给出的信息相同,在确定选择使用哪个指标时,通常是判断哪个指标值大于 1。在使用 IPC 和 CPI 比较不同计算机系统性能时,重要的是记住,IPC 值高,表明 IPC 值高的程序执行指令时所需周期数要比 IPC 值低的程序少;CPI 值高,表明 CPI 值高的程序执行指令时所需周期数要比 CPI 值低的程序多。因此,IPC 值较大,即表明系统的性能较好。

示例 某程序是由 100 条指令组成的循环,需要执行 42 次。给定系统执行程序所需时间为 16 000 个周期。在系统运行程序时,CPI 和 IPC 的值分别是多少?

解析 循环由 100 条指令组成,需要执行 42 次,所以执行指令总数是 $100 \times 42 = 4200$ 。另外,由于执行程序需要 16 000 个周期,因此 CPI 为 $16\,000 / 4200 = 3.81$ 。在计算 IPC 值时,只需将 4200 条指令除以 16 000 个周期,即可得 IPC 值为 0.26。

通常,在量度系统性能时,IPC 和 CPI 这两个指标比 MIPS 的用处更小,这是因为它们都没有包含有关系统时钟频率的任何信息,或系统在执行某个任务时需要多少条指令。当知道某程序在系统中运行时的 MIPS 值时,将 MIPS 值与程序运行过程中执行的指令数相乘,即可确定需要多长时间才能执行完程序。当知道程序在系统中运行时的 CPI 值时,将 CPI 值与程序运行过程中执行的指令数相乘,即可得到执行完程序所需要的周期总数。但必须了解每秒钟的时钟周期数,即系统的时钟频率,才能将其转换成执行程序所需的时间。

因此, CPI 和 IPC 很少用来比较实际计算机系统性能。在计算机体系结构研究中, 它们是十分常见的指标, 因为计算机体系结构研究多在仿真环境进行, 利用能模拟某种特定体系结构的程序, 估计在体系结构中执行某程序所需要的周期数。这些模拟程序通常不能预测它们模拟系统的周期时间。因此, CPI 或 IPC 通常就成为估计系统性能时所能获得的最佳指标。

1.6.3 基准测试套件

前面讨论过, 在用作量度计算机性能的指标时, MIPS、CPI、IPC 都有明显的局限性。基准测试套件 (benchmark suites) 是量度计算机性能的又一种指标, 可解决 MIPS、CPI 或 IPC 的局限性。

基准测试套件由一组程序构成, 这些程序是在系统运行的典型程序。系统在基准测试套件的得分, 根据系统执行套件中全部程序所需时间而定。由于存在许多不同的基准测试套件, 因此会生成系统在不同应用类型中的性能估计值。

最著名的基准测试套件之一是 SPEC, 它由 Standard Performance Evaluation 公司开发。本书出版时, SPEC 套件的最新版本是 SPEC CPU2000, 这是 SPEC 基准测试套件自 1989 年首次发布以来的第三个主要修订版。

基准测试套件比 MIPS、CPI 和 IPC 等指标具有更多的优点。首先, 性能结果基于总执行时间得出, 而不是基于指令执行比率。其次, 基准测试套件通过对系统在多个程序中的性能量度值求平均数, 生成平均速度估计值。这样, 与系统在任意程序中获得的 MIPS 分值相比, 系统在基准测试套件中获得的总分值更能揭示总体性能情况。此外, 许多基准测试不仅要求制造商能发布其系统在基准测试套件中的总得分, 而且要求发布系统在基准测试套件中各个程序的测试结果。这样, 当知道系统将用于某个特定应用时, 就能对各个基准测试结果直接进行比较。

1.6.4 几何平均数与算术平均数

在利用基准测试套件中包含的程序进行测试, 并对测试结果求平均数时, 许多基准测试套件采用几何平均数 (geometric), 而不是算术平均数 (arithmetic), 这是因为某个极端数值对一组数值的几何平均数影响较小, 而对该组数值的算术平均数影响较大。在采用几何平均数时, 很难通过使系统在基准测试套件中的某程序取得较好的性能测试值获得较高的基准测试套件总分值。这样就使得系统的总分值能更好地揭示系统在多数程序的性能情况。

计算 n 个数值的几何平均数的方法是先将 n 个数值相乘, 再求出乘积的 n 次方根。计算一组数值算术平均数的方法是所有数值相加, 再除以这些数值的个数。