第一图书网, tushu007.com

<<计算机组成与设计>>

图书基本信息

书名：<<计算机组成与设计>>

13位ISBN编号：9787111412373

10位ISBN编号：7111412370

出版时间：2013-2

出版时间：机械工业出版社

作者：（美）David A. Patterson,（美）John L. Hennessy

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：http://www.tushu007.com

内容概要

作者简介

作者：（美国）帕特森（Patterson D.A.） （美国）亨尼斯（Hennessy J.L.） 帕特森（Patterson D.A.）
，加州大学伯克利分校计算机科学系教授，美国国家工程研究院院士，IEEE和ACM会士，曾因成功的启发式教育方法被IEEE授予James H.Mulligan，Jr教育奖章。
他因为对RISC技术的贡献而荣获1995年IEEE技术成就奖，而在RAID技术方面的成就为他赢得了1999年IEEE ReynoldJohnson信息存储奖。
2000年他和John L.Hennessy分享了John yon Neumann奖。
亨尼斯（Hennessy J.L.），斯坦福大学校长，IEEE和ACM会士，美国国家工程研究院院士及美国科学艺术研究院院士。
Hennessy教授因为在RISC技术方面做出了突出贡献而荣获2001年的Eckert—Mauchly奖章，他也是2001年Seymour Cray计算机工程奖得主，并且和David A.Patterson分享了2000年John vonNeumann奖。

书籍目录

1 Computer Abstractions and Technology 1.1 Introduction 1.2 Below Your Program 1.3 Under the Covers 1.4 Performance 1.5 The Power Wall 1.6 The Sea Change： The Switch from Uniprocessors to Multiprocessors 1.7 Real Stuff. Ma：nufacturing and Benchmarking the AMD Opteron X4 1.8 Fallacies and Pitfalls 1.9 Concluding Remarks 1.10 Historical Perspective and Further Reading 1.11 Exercises 2 Instructions：l.anguage of the Computer 2.1 Introduction 2.2 0perations of the Computer Hardware 2.3 0perands of the Computer Hardware 2.4 Signed and Unsigned Numbers 2.5 Representing Instructions in the Computer 2.6 Logical Operations 2.7 Instructions for Making Decisions 2.8 Supporting Procedures in Computer Hardware 2.9 Communicating with People 2.10 MIPS Addressing for 32-Bit Immediates and Addresses 2.11 Parallelism and Instructions：Synchronization 2.12 Translating and Starting a Program 2.13 A C Sort Example to Put It All Together2.14 Arrays versus Pointers 2.15 Advanced Material： Compiling C and Interpreting Java 2.16 Real Stuff：ARM Instructions 2.17 Real Stuff： x86 Instructions 2.18 Fallacies and Pitfalls 2.19 Concluding Remarks 2.20 Historical Perspective and Further Reading 2.21 Exerases 3 Arithmetic for Computers 3.1 Introduction 3.2 Addition and Subtraction 3.3 Multiplication 3.4 Division 3.5 Floating Point 3.6 Parallelism and Computer Arithmetic： Associativity 3.7 Real Stuff： Floating Point in the x86 3.8 Fallacies and Pitfalls 3.9 Concluding Remarks 3.10 Historical Perspective and Further Reading 3.11 Exerases 4 The Processor 4.1 Introduction 4.2 Logic Design Conventions 4.3 Building a Datapath 4.4 A Simple Implementation Scheme 4.5 An Overview of Pipelining 4.6 Pipelined Datapath and Control 4.7 Data Hazards: Forwarding versus Stalling 4.8 Control Hazards 4.9 Exceptions 4.10 Parallelism and Advanced Instruction-Level Parallelism 4.11 Real Stuff the AMD Opteron X4 （Barcelona） Pipeline 4.12 Advanced Topic: an Introduction to Digital Design Using a Hardware Design Language to Describe and Model a Pipeline and More Pipelining Illustrations 4.13 Fallciaes and Pitfalls 4.14 Concluding Remarks 4.15 Historical Perspective and Further Reading 4.16 Exerases 5 Large and Fast: Exploiting Memory Hierarchy 6 Storage and Other I/O Topics 7 Multicores, Multiprocessors, and Clusters A Graphics and Computing GPUs A-2 B Assemblers, Linkers, and the SPIM Simulator B-2 C The Basics of Loglc Design C-2 D Mapping Control to Hardware D-2 e A Survey of RISC Architectures for Desktop, Server, and Embedded Computers E-2 G Glossary G-1 F Further Reading FR-1

章节摘录

版权页： 插图： Part of the power of the Intel x86 is, the prefixes that can modify the execution ofthe following instruction. One prefix can repeat the following instruction untila counter counts down to 0. Thus, to move data in memory, it would seem thatthe natural instruction sequence is to use move with the repeat prefix to perform 32-bit memory-to-memory moves. An alternative method, which uses the standard instructions found in all computers, is to load the data into the registers and then store the registers back tomemory. This second version of this program, with the code replicated to reduceloop overhead, copies at about 1.5 times faster. A third version, which uses thelarger floating-point registers instead of the integer registers of the x86, copies atabout 2.0 times faster than the complex move instruction. Fallacy: Write in assembly language to obtain the highest performance.At one time compilers for programming languages produced naive instructionsequences; the increasing sophistication of compilers means the gap betweencompiled code and code produced by hand is closing fast. In fact, to competewith current compilers, the assembly language programmer needs to understandthe concepts in Chapters 4 and 5 thoroughly (processor pipelining and memoryhierarchy).

编辑推荐

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:http://www.tushu007.com