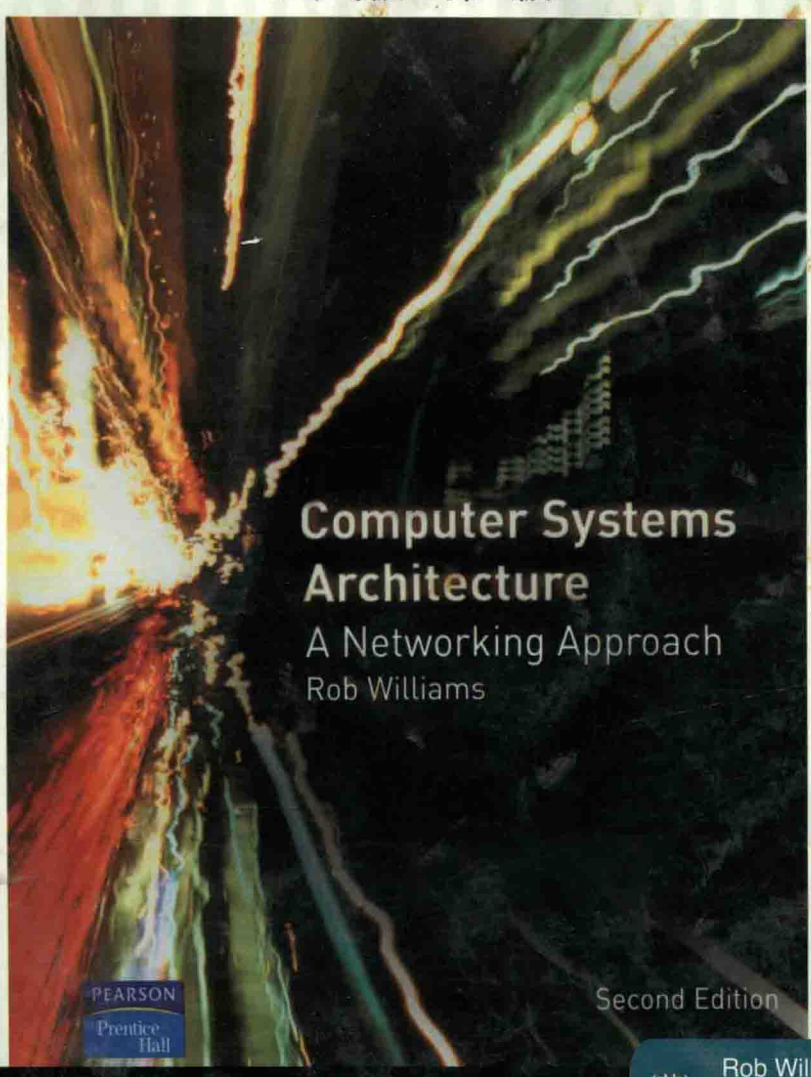
HZ BOOKS 经典原版书库

计算机系统结构

（英文版·第2版）



(英) Rob Williams 著
西英格兰大学

机械工业出版社
China Machine Press

计算机系统结构

（英文版 · 第2版）

Computer Systems Architecture A Networking Approach

(Second Edition)

本书采用自底向上的方式，依次介绍计算机系统结构的基本概念和基本内容，首先是数字逻辑电路和计算机硬件，接下来是运行于硬件之上的软件层，最后讲述通信和操作系统领域的基础知识。另外，还包含对ARM和安腾（Itanium）处理器的介绍以及数据通信延伸领域的最新知识。本书紧密联系实际，注重动手实践，利用学生感兴趣和亲身体验过的技术（如因特网、图形用户界面、移动通信等）来提高读者学习的积极性。贯穿全书，在分析系统的性能时注意将软件硬件结合起来讨论，练习题充分地展示出硬件和软件之间这种相互影响、相互依赖的基本关系。

本书适合作为高等院校计算科学及相关专业计算机系统结构的导论性教材。

本书的主要特色

- 使用实际的处理器（奔腾处理器），使学生能够在家中使用自己的机器完成绝大部分的练习作业。
- 内容组织合理，材料取自于作者自己从事教学和实验工作的真实需求。
- 介绍数据传输和通信相关的思想和概念，为联网和网络通信相关的课程打下基础。
- 每章结束后的练习均经过精心挑选，书末附有答案及注释。
- 书中用到许多现代的、商业化的实例，能够有效地激发读者的学习兴趣，并将理论与实际结合起来。

作者简介

Rob Williams 是位于英国布里斯托的西英格兰大学计算机系统技术学院院长。他在实时系统领域造诣颇深，同时还是GWE/GNE、Marconi Avionics和Nexos Office System的微处理器系统工程师。

上架指导：计算机/系统结构

ISBN 7-111-20417-4

9 787111 204176

封面设计：吴 刚

华章图书

华章网站 http://www.hzbook.com

网上购书：www.china-pub.com

投稿热线：(010) 88379604

购书热线：(010) 68995259, 68995264

读者信箱：hzjsj@hzbook.com

限中国大陆地区销售

ISBN 7-111-20417-4

定价：69.00元

计算机系统架构

(英) Rob Williams 著

Computer Systems Architecture A Networking Approach *(Second Edition)*

英文版 第2版

HZ BOOKS

出版社
ne Press

经 典 原 版 书 库

计算机系统结构

（英文版 · 第2版）

Computer Systems Architecture
A Networking Approach
(Second Edition)

(英) Rob Williams
西英格兰大学 著

机械工业出版社
China Machine Press

Rob Williams: Computer Systems Architecture: A Networking Approach, Second Edition (ISBN 13: 978-0-321-34079-5 ISBN 10: 0-321-34079-5).

Copyright © Pearson Education Limited 2001, 2006.

This edition of Computer Systems Architecture: A Networking Approach, Second Edition is published by arrangement with Pearson Education Limited. Licensed for sale in the mainland territory of the People's Republic of China only, excluding Hong Kong, Macau, and Taiwan.

本书英文影印版由英国Pearson Education培生教育出版集团授权出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

此影印版只限在中国大陆地区销售（不包括香港、澳门、台湾地区）。

版权所有，侵权必究。

本书法律顾问 北京市展达律师事务所

本书版权登记号：图字：01-2006-6508

图书在版编目（CIP）数据

计算机系统结构（英文版·第2版）/（英）威廉斯（Williams, R.）著. －北京：机械工业出版社，2007.1

（经典原版书库）

书名原文：Computer Systems Architecture: A Networking Approach, Second Edition

ISBN 7-111-20417-4

Ⅰ.计… Ⅱ.威… Ⅲ.计算机体系结构－英文 Ⅳ.TP303

中国版本图书馆CIP数据核字（2006）第139972号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）

责任编辑：迟振春

北京京北制版印刷厂印刷·新华书店北京发行所发行

2007年1月第1版第1次印刷

170mm×242mm · 47印张

定价：69.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

此为试读，需要完整PDF请访问：www.ertongbook.com

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，在“华章教育”的总规划之下出版三个系列的计算机教材：除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科

技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这三套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

电子邮件：hzjsj@hzbook.com
联系电话：（010）68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周克定	周傲英	孟小峰	岳丽华	范　明
郑国梁	施伯乐	钟玉琢	唐世渭	袁崇义
高传善	梅　宏	程　旭	程时端	谢希仁
裘宗燕	戴　葵			

Supporting resources
Visit **www.pearsoned.co.uk/williams** to find valuable online resources

Companion Website for students
- Slides and handouts for each chapter
- Laboratory worksheets
- Example scripts for class phase tests

For instructors
- Answers to test questions

For more information please contact your local Pearson Education sales representative or visit **www.pearsoned.co.uk/williams**

Preface

Writing a textbook for a subject area such as computing which changes so rapidly is a frightening experience. For this Second Edition, there was a temptation simply to create a find-and-replace editing macro which would update all numeric parameters by inserting or deleting the extra zero as appropriate! Although fundamental technical principles do not change so rapidly, it is very important to maintain a clear grasp of the speed of current processors and their likely memory size. Appreciating the possible download capacity of a broadband link can also be useful. When we estimate something, it is a valuable skill to be able to say immediately if the result is 'common sense reasonable', or plain daft. All scientists and engineers practise this skill, and so should programmers. I do remember once contributing to the loss of a commercial contract when an estimate of 6 hours for a file transfer was accepted at board level. Only later did I realize that I had forgotten to divide the number by 10, to account for Bytes/s instead of bits/s.

In this new edition, Intel's 64 bit Itanium processor has been given more space in a separate chapter because it is considered to represent a key new architecture for the post-RISC era. The terms RISC and post-RISC are used to distinguish more clearly between the original vision of simple, fast processors and their recent, more complex successors. The ARM CPU has been adopted as the example architecture for a microcontroller, using the Intel StrongARM/XScale for practical examples. More description and explanation about USB communications has also been included to reflect its growing relevance to programmers and computer users. The technology of modems has been updated to include 56k dialup and ADSL broadband equipment. The wide enthusiasm for portable devices dependent on audio and video compression techniques has stimulated a new section introducing MPEG algorithms. The sections dealing with the WWW and search engines have been overhauled and updated, using Google as the principal example.

A new chapter introducing parallel, cluster and grid configurations has been included with mention of the novel IBM Cell processor which is destined to revitalize the games console industry and, perhaps, revolutionize desktop computing.

Preparing for a new edition offers the opportunity to remove all those pernicious mistakes which somehow hang on against all our best efforts. But unfortunately it also allows for the introduction of a fresh set of errors, for which I must immediately accept the blame. I have, however, learnt the bright side of such mistakes, which is the rash of charming emails that I have received from around the world pointing out, in mostly very apologetic tones, the errors. Thanks again to those who took the trouble to get in touch with me.

I must also acknowledge the major contribution of the two commissioning editors, Keith Mansfield, for the first edition, and Simon Plumtree for the second. Without their encouragement and efforts the text would have languished on my hard disk as a troff/pic file.

rob.williams@uwe.ac.uk

Preface to the first edition

Origins of this book

This book is based on a first-year degree course called Computer Systems Architecture (uqc104s1), which I have delivered for some years at the University of the West of England (UWE), Bristol. The course has expanded, contracted and expanded again, so it has been thoroughly shaken about and reorganized several times. Many of our graduates are recruited into the growing telecommunications industry, and interest in this subject area is very high. Therefore the course, and this book, reflect the shift in interest to encompass not only applied computer science but also data communication and networking.

Students

Whereas ten years ago students came into our first year with little in the way of computing qualifications or experience, now over half our new students have already successfully completed a two-year course (A level or BTEC) in the computing field. They have enrolled on a degree course in BSc Computer Science, BSc Software Engineering or BSc Computing for Real-Time Systems. The latter, in particular, attracts technical enthusiasts who are keen to progress into exciting new subject areas such as networking, Unix, embedded systems or real-time programming. With this more demanding audience in mind, I have attempted to inject new ideas and pursue fresh directions. The danger is always that such students will become bored and withdraw their commitment if they consider the course to be 'too easy' or 'out of date'. This book attempts to acknowledge and build on their existing experience.

Another worry often voiced by students concerns the place of mathematics within the course. Well, no advanced mathematics is needed. You will only occasionally be asked to use simple algebra and arithmetic. Common sense will be your most valuable resource! Also, we will do no soldering. This is not an electronics course.

Technological change

Keeping pace with technological change is an issue for all computing courses and texts, and I have taken up this challenge by basing the book on the Pentium processor, despite its daunting complexity. Systems which seemed capable of holding their advanced position within the market-place for several years, are now overtaken within months of launch. Software tools are being developed and adopted by commercial programmers long before universities have had a chance

此为试读,需要完整PDF请访问: www.ertongbook.com

to obtain a copy. As a result, computing curricula are in need of continuous review in order to maintain their currency within the wider context. It is not uncommon for my students to return home to use their own computer systems because they are more advanced than those available in our undergraduate laboratories! But it is worrying to me how fashion-conscious the computing field has become. This can lead to rapid demotivation when the ideas and examples discussed during lectures, although academically and intellectually valid, are perceived as outdated. In this regard, I hope to have included enough contemporary material to maintain your interest!

Use of this text

We all learn differently, but the ability to use text effectively has been at the core of modern civilization for a long time. We all benefit so much from people's experience recorded on paper for others to read. Ignoring this vast resource is deliberately handicapping yourself. Life is difficult enough without conceding an unnecessary penalty! If anything, the introduction of the WWW has placed even greater literacy demands on everyone. Most Web pages presenting useful information still depend heavily on text. A picture may be worth a thousand words, but it is often the accompanying text that gives you a first glimmer of understanding.

At the start of each chapter you will see a graphic representation of the contents, modelled on the familiar Windows Explorer file browser. It enables you to see the subject context of all the chapters. It is not a contents list; rather, more an ideogram. Each chapter is supplied with a small set of questions for which answers are supplied at the back of the book. It goes without saying that you should make a serious attempt to answer the questions yourself before reading the answers! A full glossary is also included at the end of the book in an attempt to deal with the appalling jargon explosion that has taken place. We all bemoan this situation, but it does constitute part of the challenge of becoming a fully blown computer professional. It cannot simply be ignored.

Operating systems and programming languages

When presenting a course, or writing a text on computing, the choice of which language or operating system to use can be an emotional issue. Traditionally, university courses have tried to duck the question of commercial relevance and stick with the well-established academic favourites. I do find this frustrating, having once (twice?) tried to use Pascal for a commercial contract and then run up against all sorts of quite predictable problems which we had never honestly discussed with our students. With this experience in mind I have since always tried to use commercially viable tools in my teaching. The two operating systems that are universally seen throughout the world are Unix and Windows XP (in which we include Windows 2000). I expect you to have access to both systems for practical work. Linux is perfectly admirable for the Unix examples, although I have illustrated the text using Sun Microsystems' Solaris. As for languages, I assume that you are studying a programming language in parallel with this computer systems course, and so can quickly pick up the necessary level of C to understand my code fragments. I do not expect C++ expertise!

To further assist you in testing out the examples and trying the end of chapter questions, the student edition of the Microsoft Developer Studio Visual C++ *is described* in this text. The Appendix contains guidance notes to help you install the package and quickly start using it. Although we will only be using it for very elementary programming examples, you can continue to progress into C++, object-oriented programming and the use of MFCs (Microsoft Foundation Classes) if you are so minded! Source code examples and a very effective online system are provided to support your learning.

Practical course orientation

The course that you are following may be different from ours in many aspects. I plan on a 24 week schedule, splitting the work into two semesters. In this way, each chapter serves a week of the course. The order, and to some extent the content, is determined by the practical work that is carried out in the weekly lab sessions. I believe that it is principally the practical work which fully confirms technical understanding. I could never trust a software design that had not been tested by implementation! To reduce lab sessions, or decouple them from the theoretical discussion taking place in the lectures, can weaken our ability to understand and assimilate new ideas. We all learn differently. Some like to listen, some to read, and others to do. Personally, I have always found that practical activity drives home new concepts firmly and permanently.

'Listen and forget. Read and remember. Do and understand.'

For the tutor

The student readers

The typical readers that I have in mind have enrolled on a first year course in computing or a related subject area. They will be attending at least some of your weekly lectures, carrying out practical exercises, reading text books and talking to their friends about the work. They will also have regular access to the Internet, either at home or at their place of study. However, in my experience, no single one of these activities is sufficient to pass a degree course. In addition, it is unlikely that your students will only be studying computer systems and networking. In all probability there will be parallel courses in programming, systems design, mathematical methods and occasionally electronics. A successful course should help the student to integrate all these experiences and encourage cross-referencing from one discipline to another. This is precisely what this book enables them to do, since they will find links and road signs towards their other areas of study.

Our degree courses, like many, are integrated into a modular programme which offers the students some options in the second year. I find that many students require an introduction to these option subjects during the first year in order to make an informed decision. Systems administration, operating systems and networking in particular can be unknown areas for the students. This course is intended to fulfil this 'Foundation Course' function, as well as giving students a grounding in the practice and principles of computing systems.

Practical worksheets

I have made available on the Companion Web Site (www.pearsoned.co.uk/williams) samples of the weekly worksheets that I use with my students. These focus on aspects of the theoretical material which has just been presented in the lectures. Often a worksheet also serves as an introduction to the assessed coursework assignment which follows. As you can see, we still closely link student lab work sessions to their classroom experience, something which is not always possible within a modular programme.

Assessment

Because of the practical orientation of the course, I set two assessed programming assignments. In the first semester it is currently an assembler exercise using the Visual C Developer Studio. This involves accessing the PC COM port for serial communications. The second assignment is a group exercise which builds on the first. It requires the production of software to support a packet-oriented ring network, again using the PC COM port. The second programming assignment is completed in the C language, and involves protocol negotiation and cooperative working. This is intended to prepare the ground for second year courses in networking and final year courses in distributed systems. As part of the coursework assessment, I ask the students to demonstrate their code and answer a couple of questions about the structure and functionality. This does take valuable time to carry out, but in my experience is well worth the investment.

In place of a single terminal exam I prefer to set several assessed class tests. These give the student regular useful feedback, and allow for corrective action by the tutors before it is too late! The questions listed at the end of each chapter offer some preparation for the tests and benefit from summary answers provided at the end of the book. Sample test scripts are available from the publisher's secure Web site.

Acknowledgements

Thanks to Phil Naylor for supplying the example administration script listed in Chapter 13 and, more significantly, maintaining our network of Sun workstations so well. I must warmly thank my colleagues Bob Lang and Craig Duffy for doggedly reading the early drafts and offering helpful comments. I would also like to mention the many students, past and present, on our very special BSc Computing for Real-Time Systems degree, whose good humour and determination often brightened long debugging sessions. The best reward for any teacher is to see students progress and develop their technical confidence, leading to successful graduation and rewarding careers.

As the book grew, my wife's initial scepticism changed to benign toleration and then, as the text spread across the living room floor, to alarmed disbelief. But despite heroic proofreading and executive editorial efforts on her part, all errors are mine, and I hope you will send me your comments and views (rob.williams@uwe.ac.uk).

I must also give praise to Brian Kernighan for his wonderfully small pic language, which I used to draft all the line diagrams throughout the text. The original text was edited with emacs and the formatting was all carried out using groff from Richard Stallman's GNU suite. It was here that I discovered the fun of using pic to code diagrams.

However, the most critical commentary was no doubt rendered by Cassie, our morose four-year-old cat, who regularly fell sound asleep straddling the monitor on which I was editing the text.

Rob Williams
Department of Computing
University of the West of England
Bristol
July 2000

The publishers would like to express their appreciation for the invaluable advice and encouragement they have received for this book from the following academics:

Hernk Corporaal
Delft University, The Netherlands

Peter Hood
University of Huddersfield, UK

Prasant Mohaptra
Michigan State University, USA

Henk Neefs
University of Gent, Belgium

Andy Pimentel
University of Amsterdam, The Netherlands

Mike Scott
Dublin City University

Bernard Weinberg
Formerly Michigan State University, USA

Publisher's acknowledgements

We are grateful to the following for permission to reproduce copyright material:

Fig. 9.2 'Datasheet (p. 3) for a Harris 82C55A, parallel port IO chip', reproduced by permission of Intersil Corporation, © Intersil Corporation; Fig. 12.3 from Hatfield, D.J. and Gerald, J. (1971) 'Program restructuring for virtual memory', *IBM Systems Journal*, Vol. 10, No. 3, p. 189, copyright 1971 by International Business Machines Corporation, reprinted with permission of the IBM Systems Journal; Figs 15.14 and 15.15 screenshot of Netscape Communicator browser window © 2005 Netscape Communications Corporation. Used with permission. Netscape Communications has not authorized, sponsored, endorsed, or approved this publication and is not responsible for its content; Fig. 15.20 and Table 15.6 from *The Anatomy of a Large-scale Hypertextual Web Search Engine* (Brin S. and Page L., 2000) by permission of Google; Fig. 21.19 with permission of IPAQ Repair and Parts, Ratby, UK; screenshots reprinted with permission from Microsoft Corporation.

In some instances we have been unable to trace the owners of copyright material, and we would appreciate any information that would enable us to do so.

Recommended lab sessions

It is expected that the majority of the readers of this text will be students pursuing a course called Computer Science, Computing, Software Engineering, or some similar title. Such courses normally include weekly lectures and practical work in laboratories. Thus the material covered in the following chapters represents only one component of your learning experience. The following practical worksheets are available for downloading and use (from `www.pearsoned.co.uk/williams`), and have been planned to support and extend the principal ideas presented in each of the chapters. You may wish to read through them even if your own course demands that you carry out a different set of practical exercises.

Part 1

1. Videos with historic interviews and equipment
   Search engines
   Web sites with information about historic computers
   Investigate library catalogue

2. Introduction to the network
   Computing facilities, Unix and Windows NT
   Basic programmer's skills
   Revise binary numbers

3. Inside the PC
   Opening up the computer
   Identifying parts
   Reviewing component names and their function

4. Digital logic for the control unit
   Using truth tables
   Drawing simple circuits
   Understanding decoders
   Building traffic light controllers

5. Digital logic for the ALU
   Building a half adder
   Building a full adder
   Assembling a parallel adder/subtractor

6. Memory mapping
   Understanding the need for memory decoding
   Interpreting a memory map
   Writing a memory map from a hardware specification

7. Introduction to Pentium asm
   Online help
   Setting up a project on Visual Studio
   Writing and compiling C programs
   Including asm inserts

8. Subroutines
   Calling C functions
   Calling asm subroutines
   Passing parameters
   Stack frame preparation and scrubbing

9. Input and output methods using C routines
   Using the C library of IO routines
   Accessing COM1 and COM2 from C and asm

10. Inter-computer communications using RS232
    Packet data
    Flow control issues
    EOF problems

11. Memory performance
    Virtual memory swapping delays

此为试读，需要完整PDF请访问：www.ertongbook.com