国外优秀信息科学与技术系列教学用书

嵌入式软件基础

—— C语言与汇编的融合

（影印版）

FUNDAMENTALS OF EMBEDDED SOFTWARE

Where C and Assembly Meet

■ Daniel W. Lewis

PEARSON
Prentice Hall

高等教育出版社
Higher Education Press

国外优秀信息科学与技术系列教学用书

嵌入式软件基础
——C 语言与汇编的融合
（影印版）

FUNDAMENTALS OF EMBEDDED SOFTWARE
Where C and Assembly Meet

Daniel W. Lewis

PEARSON
Prentice Hall

高等教育出版社

图字：01-2003-8318 号

Fundamentals of Embedded Software: Where C and Assembly Meet
Daniel W. Lewis

原版 ISBN: 0-13-061589-7

图书在版编目（CIP）数据

嵌入式软件基础：C 语言与汇编的融合 = Fundamentals of Embedded Software：Where C and Assembly Meet／（美）刘易斯（Lewis,D.W.）．—影印本．—北京：高等教育出版社，2004.8
ISBN 7-04-014059-4

Ⅰ.嵌…　Ⅱ.刘…　Ⅲ.软件开发—高等学校—教材—英文　Ⅳ.TP311.52

中国版本图书馆 CIP 数据核字（2004）第 065485 号

出版发行	高等教育出版社	**购书热线**	010－64054588
社　　址	北京市西城区德外大街 4 号	**免费咨询**	800－810－0598
邮政编码	100011	**网　　址**	http://www.hep.edu.cn
总　　机	010－82028899		http://www.hep.com.cn
经　　销	新华书店北京发行所		
印　　刷	北京外文印刷厂		
开　　本	787×1092　1/16	**版　　次**	2004 年 8 月第 1 版
印　　张	18	**印　　次**	2004 年 8 月第 1 次印刷
字　　数	340 000	**定　　价**	35.00 元(附光盘)

Preface

How many computers are in your home? Most people might answer two or three. How many *microprocessors* are in your home? Think carefully before you answer. (*Hint:* It's a lot more than two or three! In fact, it's actually even a lot more than 10 or 20!) Today, microprocessors are embedded in almost every electronic appliance you can think of and many that you probably wouldn't. They have become pervasive—not only in our home, but in our workplace, our automobiles, airplanes, stoplights, supermarkets, cell phones—in short, in almost every aspect of our lives.

Embedded systems offer students an exciting opportunity to express their creativity. What student hasn't dreamed of designing the next new gadget that would capture the imagination of the public? Our challenge as educators is to capitalize on that excitement and channel the energy of those young minds to motivate mastery of a subject matter!

Objectives

The ultimate goal of this text is to lay a foundation that supports the multithreaded style of programming and high-reliability requirements of embedded software. Within this context, the following objectives were established:

1. To understand how data is represented at the machine level and to appreciate the consequences and limitations of those representations.
2. To master those language-specific features that are used most frequently in embedded systems, such as bit manipulation and variant access.
3. To obtain a programmer's view of processor architecture and how programming at the level of assembly is sometimes necessary or appropriate.
4. To learn about the various styles of I/O programming and, ultimately, how an event-driven approach allows one to separate data processing into a number of independent threads of computation.
5. To learn about nonpreemptive and preemptive multithreaded programming, shared resources and critical sections, and how scheduling can be used to manage system response time.
6. To reinforce basic programming skills by revisiting such topics as scope, parameter passing, recursion, and memory allocation.
7. To learn about the problems associated with shared memory objects, how shared memory is affected by memory allocation, and what programming practices can be used to minimize the occurrence of shared memory.

Intended Audience

This text is intended to serve as the basis for a sophomore-level course in a computer science, computer engineering, or electrical engineering curriculum. Such a course is envisioned as a replacement for the traditional course on computer organization and assembly language programming.

The text presents assembly the way it is most commonly used in practice—to implement small, fast, or special-purpose routines called from a main program written in a high-level language such as C. It thus covers processor organization and assembly language only from a "need-to-know" point of view, rather than as a primary objective. This approach affords time within both the text and the course to cover assembly in the context of embedded software. As a result, students not only learn that assembly still has an important role to play, but their discovery of multithreaded programming, preemptive and nonpreemptive systems, shared resources, and scheduling helps sustain their interest, feeds their curiosity, and strengthens their preparation for subsequent courses on operating systems, real-time systems, networking, and microprocessor-based design.

At most institutions, the introductory programming sequence (CS1, CS2) is no longer taught in a procedural programming language such as C or Pascal; rather, the popular approach is to now use an object-oriented programming language such as C++ or Java. Despite the change, it is not uncommon to find one or more upper division courses still using a procedural language, nor is it uncommon to find such languages still in use in industry. At the author's institution, we solved this paradox by redesigning our traditional assembly-language course around the material in this book; it not only created room in an already packed curriculum to cover the procedural approach and to introduce the popular topic of embedded systems, but it has also offered an opportunity to strengthen student comprehension of parameter passing, scope, and memory allocation schemes that they were first introduced to in CS1 and CS2.

The text assumes that students already know how to program in C, C++, or Java, and that the similarity among the low-level syntax of those languages makes it relatively easy to move to C from either C++ or Java. Rather than covering C in excruciating detail, the text emphasizes those features of C that are employed more frequently in embedded applications, and introduces the procedural style through examples and programming assignments that include large amounts of prewritten source code. In principle, the only absolute prerequisite is thus a CS1 course that uses C, C++, or Java. However, additional programming maturity such as acquired from a CS2 course on data structures is strongly recommended.

Programming Assignments and the CD-Rom

The text is complemented by a collection of programming assignments described in Appendix D. Given that the text is aimed at sophomores, the assignments are intended primarily to illustrate a topic from the text, rather than as extended programming projects. As such, most of the source code for each assignment is provided on the CD-Rom and students are asked to focus only on those parts that relate directly to the topic. For example, the last three assignments provide complete source code to graphically demonstrate problems associated with shared resources, priority inversion, and deadlock, and require the student to correct these problems by modifying specific parts of the code by using strategies presented in the text.

The programming assignments and when they should be scheduled relative to material in the text is summarized as follows:

Programming Assignment	Relevant Chapter	Comment
1	n/a	Introduction to C and the DJGPP compiler; requires no knowledge of material in the text, and may be scheduled during the first week of the course.
2	2	Fixed-precision real numbers in C.
3	3	Macros and packed operands in C.
4	n/a	makefiles: Typically assigned while studying Chapter 4, but has no direct relationship to that material.
5 and 6	5	Assembly-language programming.
7	6	Interrupt-driven I/O in assembly.
8	7	Multi-threaded programming with a nonpreemptive kernel.
9	7	Preemptive kernels, shared resources, and semaphores.
10 and 11	8	Scheduling problems (priority inversion and deadlock).

On the CD-Rom you'll also find all the software tools needed to develop embedded applications under Microsoft Windows 9X, 2000 and NT: the DJGPP port of the gnu C compiler and linker, a compatible assembler and run-time libraries. A boot loader is provided to load (and execute) the embedded application into memory from diskette. Where possible, source code for each of these tools has also been included. Directions for using each of these tools are found at the beginning of Appendix D.

Choice of Platform

The text uses the ubiquitous PC as a platform for learning about processor architecture, assembly language, and embedded software. Two primary factors motivated this choice: (1) Students will ultimately encounter the Intel architecture due to its dominance within the PC market, and (2) choosing to build embedded applications on the PC has the added benefit of allowing instructors to offer an associated laboratory component without investing in specialized single board computers.

Most assembly-language programming textbooks for the Intel processor cover the original "real" mode of the 8088. However, the protected mode of the 386 and later Intel processors is much more representative of modern architecture and is actually much easier to program when configured to use a "flat" memory model. Although real mode is covered briefly in the text, the emphasis is on protected mode and is used in all the of assembly-language examples.

One should not construe that a PC is the best (or even an appropriate) platform for building *actual* embedded systems. Most embedded applications have no need for many of the PC's standard peripherals (e.g., display, keyboard, disk), and the power-on boot procedure in the ROM BIOS doesn't even support diskless applications. Trying to write initialization code to replace the BIOS is difficult at best, because it requires knowledge of chipset-dependent features that vary from PC to PC and which are often proprietary.

Acknowledgments

This book and the software on the companion CD would not have been possible without the efforts of a number of people. Of them all, I owe a special thanks to the students and teaching assistants of COEN 20 at Santa Clara University who suffered through the rough early versions of the manuscript and helped to debug the libepc library routines.

I also owe a debt of gratitude to many of my colleagues: To Qiang Li and Neil Quinn, who endured the unenviable task of teaching from my materials and who graciously suggested a number of improvements to the text. To Hal Brown and Vasu Alagar, who helped me develop a mathematical representation of fixed-point multiplication of real numbers that led to an efficient straight-line implementation in assembly and an explanation of the algorithm in section 2.3.3 that students find relatively easy to understand. To Darren Atkinson, who spent the better part of two days in my office tracking down an innocuous little software bug that only appeared when I upgraded to a more recent version of the compiler. And a hearty thanks to early reviewers of my manuscript, whose comments helped to strengthen the final book. They are Dr. Walter Higgins of Arizona State University, Dr. Karram Mossaad of the University of Texas at Austin, and Dr. Dana Lasher of North Carolina State University. Special thanks to Dr. Lasher, who contributed the telephone call analogy that appears in the opening paragraph of section 6.4 on Interrupt-Driven I/O.

We should *all* be grateful to those unselfish programmers who have created professional quality software tools and then made them readily available for anyone to use: To DJ Delorie for his DJGPP port of the GNU C compiler, to Simon Tatham and Julian Hall for their NASM assembler, to Jean Labrosse for his μC/OS-II preemptive real-time kernel, and to Dennis Saunders of MIX Software for their Multi-C non-preemptive real-time kernel.

Daniel W. Lewis

Prentice-Hall companion website:
www.prenhall.com/divisions/ems/app/lewis

出 版 说 明

20 世纪末，以计算机和通信技术为代表的信息科学和技术对世界经济、科技、军事、教育和文化等产生了深刻影响。信息科学技术的迅速普及和应用，带动了世界范围信息产业的蓬勃发展，为许多国家带来了丰厚的回报。

进入 21 世纪，尤其随着我国加入 WTO，信息产业的国际竞争将更加激烈。我国信息产业虽然在 20 世纪末取得了迅猛发展，但与发达国家相比，甚至与印度、爱尔兰等国家相比，还有很大差距。国家信息化的发展速度和信息产业的国际竞争能力，最终都将取决于信息科学技术人才的质量和数量。引进国外信息科学和技术优秀教材，在有条件的学校推动开展英语授课或双语教学，是教育部为加快培养大批高质量的信息技术人才采取的一项重要举措。

为此，教育部要求由高等教育出版社首先开展信息科学和技术教材的引进试点工作。同时提出了两点要求，一是要高水平，二是要低价格。在高等教育出版社和信息科学技术引进教材专家组的努力下，经过比较短的时间，第一批由教育部高等教育司推荐的 20 多种引进教材已经陆续出版。这套教材出版后受到了广泛的好评，其中有不少是世界信息科学技术领域著名专家、教授的经典之作和反映信息科学技术最新进展的优秀作品，代表了目前世界信息科学技术教育的一流水平，而且价格也是最优惠的，与国内同类自编教材相当。这套教材基本覆盖了计算机科学与技术专业的课程体系，体现了权威性、系统性、先进性和经济性等特点。

目前，教育部正在全国 35 所高校推动示范性软件学院的建设，这也是加快培养信息科学技术人才的重要举措之一。为配合软件学院的教学工作，结合各软件学院的教学计划和课程设置，高等教育出版社近期聘请有关专家和软件学院的教师遴选推荐了一批相应的原版教学用书，正陆续组织出版，以方便各软件学院开展双语教学。

我们希望这些教学用书的引进出版，对于提高我国高等学校信息科学技术的教学水平，缩小与国际先进水平的差距，加快培养一大批具有国际竞争力的高质量信息技术人才，起到积极的推动作用。同时我们也欢迎广大教师和专家们对我们的教材引进工作提出宝贵的意见和建议。联系方式：hep.cs@263.net。

高等教育出版社
二〇〇二年九月

The greatest challenge to writing a book is finding the time, and working on it at home in the evenings and on weekends often becomes an unavoidable way of life for several months. Throughout it all my wife and children have been more understanding, patient, and supporting than I would ever have expected. Thank you. This book is dedicated to you.

Contents

C H A P T E R 1

Introduction

1.1 WHAT IS AN EMBEDDED SYSTEM?

Embedded systems are electronic devices that incorporate microprocessors within their implementations. The main purposes of the microprocessor are to simplify system design and provide flexibility. Having a microprocessor in the device means that removing bugs, making modifications, or adding new features are only matters of rewriting the software that controls the device. Unlike PCs, however, embedded systems may not have a disk drive and so the software is often stored in a read-only memory (ROM) chip; this means that modifying the software requires either replacing or "reprogramming" the ROM.

As Table 1–1 indicates, embedded systems are found in a wide range of application areas. Originally they were used only for expensive industrial-control applications, but as technology brought down the cost of dedicated processors, they began to appear in moderately expensive applications such as automobiles, communications and office equipment, and televisions. Today's embedded systems are so inexpensive that they are used in almost every electronic product in our life.

In many cases we're not even aware that a computer is present and so don't realize just how pervasive they have become. For example, although the typical family may own only one or two personal computers, the number of embedded computers found within their home and cars and among their personal belongings is much greater.

What is often surprising is that embedded processors account for virtually 100% of worldwide microprocessor production! For every microprocessor produced for use

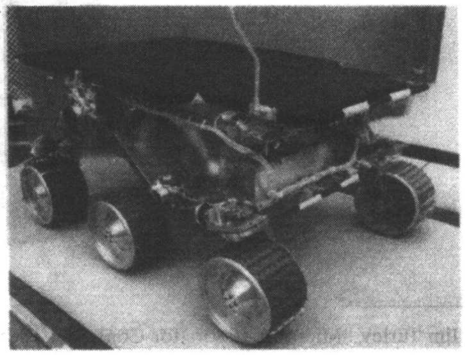

FIGURE 1–1 NASA's Mars Sojourner Rover used an Intel 80C85 8-bit microprocessor. *Courtesy of NASA/JPL.*