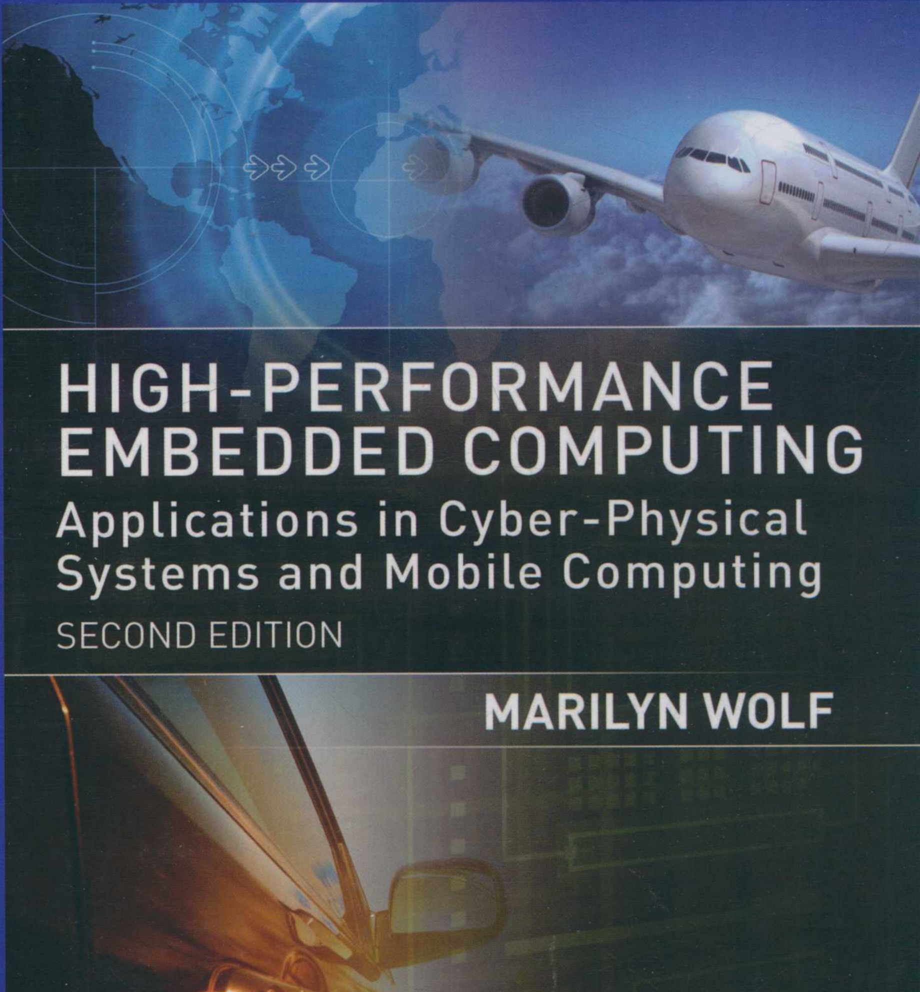
华章教育

经典原版书库

高性能嵌入式计算

[美] 玛里琳·沃尔夫 (Marilyn Wolf) 著

(英文版·第2版)



HIGH-PERFORMANCE EMBEDDED COMPUTING

Applications in Cyber-Physical Systems and Mobile Computing

SECOND EDITION

MARILYN WOLF

High-Performance Embedded Computing

Applications in Cyber-Physical Systems and Mobile Computing (Second Edition)

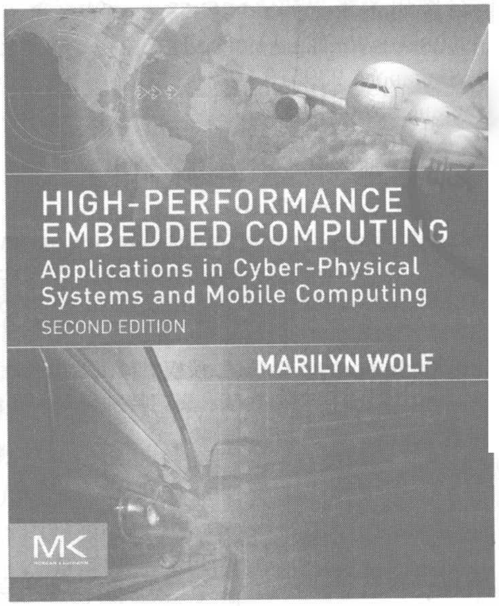
机械工业出版社
China Machine Press

经 典 原 版 书 库

高性能嵌入式计算

（英文版 · 第2版）

High-Performance Embedded Computing
Applications in Cyber-Physical Systems and Mobile Computing (Second Edition)


HIGH-PERFORMANCE EMBEDDED COMPUTING
Applications in Cyber-Physical Systems and Mobile Computing
SECOND EDITION
MARILYN WOLF
MK

[美] 玛里琳 · 沃尔夫 (Marilyn Wolf) 著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

高性能嵌入式计算（英文版·第2版）/（美）沃尔夫（Wolf, M.）著. —北京：机械工业出版社，2015.4
（经典原版书库）
书名原文：High-Performance Embedded Computing: Applications in Cyber-Physical Systems and Mobile Computing, Second Edition

ISBN 978-7-111-49930-5

I. 高… II. 沃… III. 微型计算机－系统设计－英文 IV. TP360.21

中国版本图书馆 CIP 数据核字（2015）第 073974 号

本书版权登记号：图字：01-2015-1270

出版发行：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	责任校对：董纪丽
印　　刷：三河市宏图印务有限公司	版　　次：2015 年 5 月第 1 版第 1 次印刷
开　　本：186mm×240mm　1/16	印　　张：31.25
书　　号：ISBN 978-7-111-49930-5	定　　价：79.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：(010) 88378991　88361066	投稿热线：(010) 88379604
购书热线：(010) 68326294　88379649　68995259	读者信箱：hzjsj@hzbook.com

出版者的话

文艺复兴以来，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；也正是这样的优势，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，还揭示了学术的源变，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和发展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到积极的推动作用，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公司较早意识到“出版要为教育服务”。自1998年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与Pearson，McGraw-Hill，Elsevier，MIT，John Wiley & Sons，Cengage等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出Andrew S. Tanenbaum，Bjarne Stroustrup，Brain W. Kernighan，Dennis Ritchie，Jim Gray，Afred V. Aho，John E. Hopcroft，Jeffrey D. Ullman，Abraham Silberschatz，William Stallings，Donald E. Knuth，John L. Hennessy，Larry L. Peterson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、研究及珍藏。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相助，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，有的还专门为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，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。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

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。随着计算机科学与技术专业学科建设的不断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将步入一个新的阶段，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。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：

华章网站：www.hzbook.com
电子邮件：hzjsj@hzbook.com
联系电话：(010) 88379604
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

HZ BOOKS
华章教育

华章科技图书出版中心

Preface to the Second Edition

In this second edition, I tried to take a fresh look at the field of embedded computing and the range of techniques we now can apply to the design of high-performance embedded systems. A stunning range of applications now use embedded multiprocessors and complex software stacks: smartphones, airplanes, cars, electric power equipment, and the list goes on. These complex applications require sophisticated hardware, software, and design methodologies.

I updated every chapter; some required more revision than others. I also added a new chapter on cyber-physical systems. CPS has emerged since the publication of the first edition as a synthesis of control theory and embedded computing. While CPS leverages many of the techniques described in the other chapters, a number of new results apply to the design of cyber-physical systems. I tried to both outline those results and to highlight their relationship to results described in the other chapters.

My editors, Todd Green and Nate McFadden, gave me invaluable support and advice. I would like to thank the anonymous reviewers for their thoughtful and helpful comments. Any faults in this book, conceptual, technical, or stylistic, are all mine.

Marilyn Wolf
Atlanta GA

Preface to the First Edition

This book's goal is to provide a frame of reference for the burgeoning field of high-performance embedded computing. Embedded computers have moved well beyond the early days of 8-bit microcontrollers. Today, embedded computers are organized into multiprocessors that run millions of lines of code. They do so in real time and at very low power levels. In order to properly design such systems, a large and growing body of research has developed to answer questions about the characteristics of embedded hardware and software. These are real systems—aircraft, cell phones, and digital television all rely on high-performance embedded systems. We understand quite a bit about how to design such systems, but we also have a great deal more to understand.

Real-time control was actually one of the first uses of computers—Chapter 1 mentions the MIT Whirlwind computer, developed in the 1950s for weapons control. But the microprocessor moved embedded computing to the front burner as an application area for computers. Although sophisticated embedded systems were in use by 1980, embedded computing as an academic field didn't emerge until the 1990s. Even today, many traditional computer science and engineering disciplines study embedded computing topics without being fully aware of related work being done in other disciplines.

Embedded computers are very widely used, with billions sold each year. A huge number of practitioners design embedded systems—a half-million programmers design embedded software. Although embedded systems vary widely in their details, there are common principles that apply to the field of embedded computing. Some of those principles were discovered decades ago while others are just being developed today. The development of embedded computing as a research field has helped to move embedded system design from a craft to a discipline, a move that is entirely appropriate given the important (sometimes safety-critical) tasks entrusted to embedded computers.

One reasonable question to ask about this field is how it differs from traditional computer systems topics, such as client-server systems or scientific computing. Are we just applying the same principles to smaller systems or do we need to do something new? I believe that embedded computing, though it makes use of many techniques from computer science and engineering, poses some unique challenges. First, most if not all embedded systems must perform tasks in real time. This requires a major shift in thinking for both software and hardware designers. Second, embedded computing also puts a great deal of emphasis on power and energy consumption. While power is important in all aspects of computer systems, embedded applications tend to be closer to the edge of the energy-operation envelope than many general-purpose systems. All this leads to embedded systems being more heavily engineered to meet a particular set of requirements than are general-purpose computers, which, after all, are designed for general use.

This book assumes that you, the reader, are familiar with the basics of embedded hardware and software, such as might be found in *Computers as Components*. This book builds on those foundations to study a range of advanced topics. In selecting topics to cover, I tried to identify topics and results that are unique to embedded computing. I did include some background material from other disciplines to help set the stage for a discussion of embedded systems problems.

Here is a brief tour through the book:

Chapter 1 provides some important background for the rest of the book. It tries to define the set of topics that are at the center of embedded computing. It looks at methodologies and design goals. We survey models of computation, which serve as a frame of reference for the characteristics of applications. It also surveys several important applications that rely on embedded computing in order to provide some terminology that can be used throughout the book.

Chapter 2 looks at processors. We consider several different styles of processors that are used in embedded systems. We consider techniques for tuning the performance of a processor, such as voltage scaling. We consider the role of the processor memory hierarchy in embedded CPUs. We look at techniques used to optimize embedded CPUs, such as code compression and bus encoding. We consider techniques for simulating processors.

Chapter 3 studies programs. The back end of the compilation process, which helps determine the quality of the code, is the first topic. We spend a great deal of time on memory system optimizations since memory behavior is a prime determinant of both performance and energy consumption. We consider performance analysis including both simulation and worst-case execution time analysis. We also consider how models of computing are reflected in programming models and languages.

Chapter 4 moves up to multiple-process systems. We study and compare scheduling algorithms, including the interaction between language design and scheduling mechanisms. We evaluate operating system architectures and the overhead incurred by the operating system. We also consider methods for verifying the behavior of multiple process systems.

Chapter 5 concentrates on multiprocessor architectures. We consider both tightly coupled multiprocessors and the physically distributed systems used in vehicles. We consider architectures and their components: processors, memory, and networks. We also look at methodologies for multiprocessor design.

Chapter 6 looks at software for multiprocessors. We consider scheduling algorithms for multiprocessors. We also study middleware architectures for dynamic resource allocation in multiprocessors.

Chapter 7 concentrates on hardware/software co-design. We study different models that have been used to characterize embedded applications and target architectures. We study a wide range of algorithms for co-synthesis and we compare the models and assumptions used by these algorithms.

Hopefully this book covers at least most of the topics of interest to a practitioner and student of advanced embedded computing systems. There were some topics for which I could find surprisingly little work in the literature, software testing for embedded systems being a prime example. I tried to find representative articles of the major approaches to each problem. I am sure that I have failed in many cases to adequately represent a particular problem, for which I apologize.

This book is about embedded computing; it touches upon but does not exhaustively cover several related fields:

Applications Embedded systems are designed to support applications like multimedia, communications, etc. Chapter 1 introduces some basic concepts about a few applications and understanding something about the application domain is important. But an in-depth look at these fields is best left to others.

VLSI Although systems-on-chips are an important medium for embedded systems, they are not the only medium. Automobiles, airplanes, and many other important systems are controlled by distributed embedded networks.

Hybrid systems The field of hybrid systems studies the interactions between continuous and discrete systems. This is an important and interesting area, and many embedded systems can make use of hybrid system techniques, but hybrid systems deserve their own book.

Software engineering Software design is a rich field that provides critical foundations but leave many questions specific to embedded computing unanswered.

I would like to thank all the people who have helped me with this book: Robert P. Adler (Intel), Brian Butler (Qualcomm), Alain Darte (CNRS), Babak Falsafi (CMU), Ran Ginosar (Gen-Gurion University), John Glossner (Sandbridge), Graham Hellestrand (VaSTSystems), Paolo Ienne (EPFL), Masaharu Imai (Osaka University), Irwin Jacobs (Qualcomm), Axel Jantsch (KTH), Ahmed Jerraya (TIMA), Lizy Kurian John (UT Austin), Christoph Kirsch (University of Salzburg), Haris Lekatsas (NEC), Frank Mueller (NCSU), Pierre Paulin (ST Microelectronics), Laura Pozzi (University of Lugano), Chris Rowen (Tensukuca), Rob Rutenbar (CMU), Deepu Talla (TI), David Whalley (FSU), Jiang Xu (Sandbridge), and Shengqi Yang (Intel).

I greatly appreciate the support, guidance, and encouragement given by my editor Nat McFadden, as well as the reviewers he worked with. The review process has helped identify the proper role of this book and Nat provided a steady stream of insightful thoughts and comments. I'd also like to thank my long-standing editor at Morgan Kaufman, Denise Penrose, who shepherded this book from the beginning.

I'd also like to express my appreciation to digital libraries, particularly those of the IEEE and ACM. I am not sure that this book would have been possible without them. If I had to find all the papers that I studied in a bricks-and-mortar library, I would have rubbery legs from walking through the stacks, tired eyes, and thousands of paper cuts. With the help of digital libraries, I only have the tired eyes.

And for the patience of Nancy and Alec, my love.

Wayne Wolf
Princeton, New Jersey

Acknowledgments

Contents

CHAPTER

1

Embedded Computing

CHAPTER POINTS

- Fundamental problems in embedded computing
- Cyber-physical systems and embedded computing
- Design methodologies and system modeling for embedded systems
- Models of computation
- Reliability and security
- Consumer electronics

1.1 The landscape of high-performance embedded computing

High-performance embedded computing systems are everywhere: your cell phone, your car, the medical equipment your doctor uses. Not only do they require lots of computation, but they must meet quantifiable goals: real-time performance, not just average performance; power/energy consumption; and cost. The fact that we have quantifiable goals makes the design of embedded computing systems a very different experience than the design of general-purpose computing systems, in which we cannot predict the uses to which the computer will be put.

When we try to design computer systems to meet these sorts of quantifiable goals, we quickly come to the conclusion that no one system is best for all applications. Different requirements lead us to different trade-offs between performance and power, hardware and software, and so on. We must create different implementations to meet the needs of a family of applications. Solutions should be programmable enough to make the design flexible and long-lived, but not provide unnecessary flexibility that would detract from meeting the system requirements.

General-purpose computing systems separate the design of hardware and software, but in embedded computing systems we can simultaneously design the hardware and software. We often find that we can solve a problem by hardware means, software means, or a combination of the two. These solutions may have different trade-offs; the larger design space afforded by joint hardware/software design allows us to find better solutions to design problems.

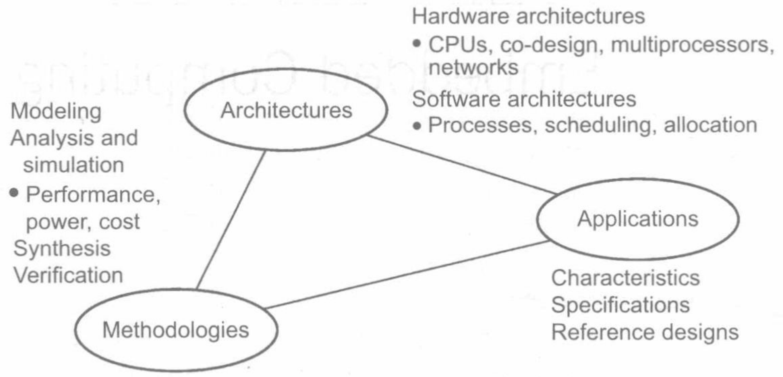


FIGURE 1.1

Aspects of embedded system design.

Architectures, applications, methodologies

As illustrated in Figure 1.1, the study of embedded system design properly takes into account three aspects of the field: **architectures**, **applications**, and **methodologies**. Compared to the design of general-purpose computers, embedded computer designers rely much more heavily on both methodologies and a basic knowledge of applications. Let's consider these aspects one at a time.

Architectures

Because embedded system designers work with both hardware and software, they must study architectures broadly speaking, including hardware, software, and the relationships between the two. Hardware architecture problems may range from special-purpose hardware units as created by hardware/software co-design, micro-architectures for processors, multiprocessors, or networks of distributed processors. Software architectures determine how we can take advantage of parallelism and non-determinism to improve performance and lower cost.

Applications

Understanding your application is critical to getting the most out of an embedded computing system. We can use the characteristics of the application to optimize our design. This can be an advantage that lets us perform many powerful optimizations that would not be possible in a general-purpose system. But it also means that we must understand the application well enough to be able to take advantage of its characteristics and avoid creating problems for system implementers.

Methodologies

Methodologies play an especially important role in embedded computing. Not only must we design many different types of embedded systems, but we want to be able to conduct a design process that is reliable and predictable. The cost of the design process itself is often a significant component of the total system cost. Methodologies, which may combine tools and manual steps, codify our knowledge on how to design systems. Methodologies help us make large and small design decisions.

The designers of general-purpose computers stick to a more narrowly defined hardware design methodology based on elements such as standard benchmarks like inputs, tracing, and simulation. The changes to the processor are generally made by hand and may be the result of invention. Embedded computing system designers need more complex methodologies because their system design encompasses both hardware