HZ BOOKS

经 典 原 版 书 库

ELSEVIER

嵌入式计算

体系结构、编译器和工具的VLIW方法

（英文版）

ELSEVIER

Joseph A. Fisher • Paolo Faraboschi • Cliff Young

Embedded Computing

A VLIW Approach to Architecture, Compilers, and Tools

Joseph A. Fisher
（美） Paolo Faraboschi 著
Cliff Young

机械工业出版社
China Machine Press

经 典 原 版 书 库

嵌入式计算

体系结构、编译器和工具的VLIW方法

（英文版）

Embedded Computing

A VLIW Approach to Architecture, Compilers, and Tools

（美） Joseph A. Fisher
Paolo Faraboschi 著
Cliff Young

机械工业出版社
China Machine Press

本书版权登记号：图字：01-2006-3121

图书在版编目（CIP）数据

嵌入式计算：体系结构、编译器和工具的VLIW方法（英文版）/（美）费希尔（Fisher, J. A.）等著. –北京：机械工业出版社，2006. 9

（经典原版书库）

书名原文：Embedded Computing: A VLIW Approach to Architecture, Compilers, and Tools

ISBN 7-111-19771-2

Ⅰ. 嵌…　Ⅱ. 费…　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（2006）第096575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

北京诚信伟业印刷有限公司印刷 · 新华书店北京发行所发行

2006年9月第1版第1次印刷

170mm × 242mm · 44.25印张

定价：89.00元

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，还揭櫫了学术的源变，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

To my wife Elizabeth, our children David and Dora,
and my parents, Harry and the late Susan Fisher.
And to my friend and mentor, Martin Davis.
Josh Fisher

To the memory of my late parents Silvio and Gina,
to my wife Tatiana and our daughter Silvia.
Paolo Faraboschi

To the women of my family:
Yueh-Jing, Dorothy, Matilda, Joyce, and Celeste.
Cliff Young

To Bob Rau, a VLIW pioneer and true visionary,
and a wonderful human being.
We were privileged to know and work with him.
The Authors

Praise for *Embedded Computing: A VLIW Approach to Architecture, Compilers and Tools*

There is little doubt that embedded computing is the new frontier of computer research. There is also a consensus that VLIW technology is extremely powerful in this domain. This book speaks with an authoritative voice on VLIW for embedded with true technical depth and deep wisdom from the pioneering experiences of the authors. This book will find a place on my shelf next to the classic texts on computer architecture and compiler optimization. It is simply that good.

Tom Conte Center for Embedded Systems Research, North Carolina State University

Written by one of the field's inventors with his collaborators, this book is the first complete exposition of the VLIW design philosophy for embedded systems. It can be read as a stand-alone reference on VLIW—a careful treatment of the ISA, compiling and program analysis tools needed to develop a new generation of embedded systems—or as a series of design case studies drawn from the authors' extensive experience. The authors' style is careful yet informal, and the book abounds with "flames," debunked "fallacies" and other material that engages the reader in the lively interplay between academic research and commercial development that has made this aspect of computer architecture so exciting. Embedded Computing: A VLIW Approach to Architecture, Compilers, and Tools *will certainly be the definitive treatment of this important chapter in computer architecture.*

Richard DeMillo Georgia Institute of Technology

This book does a superb job of laying down the foundations of VLIW computing and conveying how the VLIW principles have evolved to meet the needs of embedded computing. Due to the additional attention paid to characterizing a wide range of embedded applications and development of an accompanying toolchain, this book sets a new standard both as a reference and a text for embedded computing.

Rajiv Gupta The University of Arizona

A wealth of wisdom on a high-performance and power-efficient approach to embedded computing. I highly recommend it for both engineers and students.

Norm Jouppi HP Labs

Josh, Paolo, and Cliff have devoted most of their professional lives to developing and advancing the fundamental research and use of VLIW architectures and instruction level parallelism. They are also system-builders in the best and broadest sense of the term. This book offers deep insights into the field, and highlights the power of these technologies for use in the rapidly expanding field of high performance embedded computing. I believe this book will become required reading for anyone working in these technologies.

Dick Lampman HP Labs

Embedded Computing *is a fabulous read, engagingly styled, with generous research and practical perspective, and authoritative, since Fisher has been responsible for this paradigm of simultaneously engineering the compiler and processor. Practicing engineers—both architects and embedded system designers—will find the techniques they will need to achieve the substantial benefits of VLIW-based systems. Instructors will value the rare juxtaposition of advanced technology with practical deployment examples, and students will enjoy the unusually interesting and mind-expanding chapter exercises.*

Richard A. Lethin Reservoir Labs and Yale University

One of the strengths of this book is that it combines the perspectives of academic research, industrial development, as well as tool building. While its coverage of embedded architectures and compilers is very broad, it is also deep where necessary. Embedded Computing *is a must-have for any student or practitioner of embedded computing.*

Walid Najjar University of California, Riverside

About the Authors

JOSEPH A. FISHER is a Hewlett-Packard Senior Fellow at HP Labs, where he has worked since 1990 in instruction-level parallelism and in custom embedded VLIW processors and their compilers. Josh studied at the Courant Institute of NYU (B.A., M.A., and then Ph.D. in 1979), where he devised the trace scheduling compiler algorithm and coined the term *instruction-level parallelism*. As a professor at Yale University, he created and named VLIW architectures and invented many of the fundamental technologies of ILP. In 1984, he started Multiflow Computer with two members of his Yale team. Josh won an NSF Presidential Young Investigator Award in 1984, was the 1987 Connecticut Eli Whitney Entrepreneur of the Year, and in 2003 received the ACM/IEEE Eckert-Mauchly Award.

PAOLO FARABOSCHI is a Principal Research Scientist at HP Labs. Before joining Hewlett-Packard in 1994, Paolo received an M.S. (Laurea) and Ph.D. (Dottorato di Ricerca) in electrical engineering and computer science from the University of Genoa (Italy) in 1989 and 1993, respectively. His research interests skirt the boundary of hardware and software, including VLIW architectures, compilers, and embedded systems. More recently, he has been looking at the computing aspects of demanding content-processing applications. Paolo is an active member of the computer architecture community, has served in many program committees, and was Program Co-chair for MICRO (2001) and CASES (2003).

CLIFF YOUNG works for D. E. Shaw Research and Development, LLC, a member of the D. E. Shaw group of companies, on projects involving special-purpose, high-performance computers for computational biochemistry. Before his current position, he was a Member of Technical Staff at Bell Laboratories in Murray Hill, New Jersey. He received A.B., S.M., and Ph.D. degrees in computer science from Harvard University in 1989, 1995, and 1998, respectively.

Foreword

Bob Colwell, R & E Colwell & Assoc. Inc.

There are two ways to learn more about your country: you can study it directly by traveling around in it or you can study it indirectly by leaving it. The first method yields facts and insights directly in context, and the second by contrast.

Our tradition in computer engineering has been to seldom leave our neighborhood. If you want to learn about operating systems, you read an OS book. For multiprocessor systems, you get a book that maps out the MP space.

The book you are holding in your hands can serve admirably in that direct sense. If the technology you are working on is associated with VLIWs or "embedded computing," clearly it is imperative that you read this book.

But what pleasantly surprised me was how useful this book is, even if one's work is not VLIW-related or has no obvious relationship to embedded computing. I had long felt it was time for Josh Fisher to write his magnum opus on VLIWs, so when I first heard that he and his coauthors were working on a book with VLIW in the title I naturally and enthusiastically assumed this was it. Then I heard the words "embedded computing" were also in the title and felt considerable uncertainty, having spent most of my professional career in the general-purpose computing arena. I thought embedded computing was interesting, but mostly in the same sense that studying cosmology was interesting: intellectually challenging, but what does it have to do with me?

I should have known better. I don't think Josh Fisher can write boring text. He doesn't know how. (I still consider his "Very Long Instruction Word Architectures and the ELI-512" paper from ISCA-10 to be the finest conference publication I have ever read.) And he seems to have either found like-minded coauthors in Faraboschi and Young or has taught them well, because *Embedded Computing: A VLIW Approach to Architecture, Tools and Compilers* is enthralling in its clarity and exhilarating in its scope. If you are involved in computer system design or programming, you must still read this book, because it will take you to places where the views are spectacular, including those looking over to where you usually live. You don't necessarily have to agree with every point the authors make, but you *will* understand what they are trying to say, and they *will* make you think.

One of the best legacies of the classic Hennessy and Patterson computer architecture textbooks is that the success of their format and style has encouraged more books like theirs. In *Embedded Computing: A VLIW Approach to Architecture, Tools and Compilers*, you will find the pitfalls, controversies, and occasional opinion sidebars that made

H&P such a joy to read. This kind of technical exposition is like vulcanology done while standing on an active volcano. Look over there, and see molten lava running under a new fissure in the rocks. Feel the heat; it commands your full attention. It's immersive, it's interesting, and it's immediate. If your Vibram soles start melting, it's still worth it. You probably needed new shoes anyway.

I first met Josh when I was a grad student at Carnegie-Mellon in 1982. He spent an hour earnestly describing to me how a sufficiently talented compiler could in principle find enough parallelism, via a technique he called trace scheduling, to keep a really wild-looking hardware engine busy. The compiler would speculatively move code all over the place, and then invent more code to fix up what it got wrong. I thought to myself "So *this* is what a lunatic looks like up close. I hope he's not dangerous." Two years later I joined him at Multiflow and learned more in the next five years than I ever have, before or since.

It was an honor to review an early draft of this book, and I was thrilled to be asked to contribute this foreword. As the book makes clear, general-purpose computing has traditionally gotten the glory, while embedded computing quietly keeps our infrastructure running. This is probably just a sign of the immaturity of the general-purpose computing environment (even though we "nonembedded" types don't like to admit that). With general-purpose computers, people "use the computer" to do something. But with embedded computers, people accomplish some task, blithely and happily unaware that there's a computer involved. Indeed, if they had to be conscious of the computer, their embedded computers would have already failed: antilock brakes and engine controllers, for instance. General-purpose CPUs have a few microarchitecture performance tricks to show their embedded brethren, but the embedded space has much more to teach the general computing folks about the bigger picture: total cost of ownership, who lives in the adjacent neighborhoods, and what they need for all to live harmoniously. This book is a wonderful contribution toward that evolution.

Bob Colwell
June 17, 2004

Preface

Welcome to our book. We hope you enjoy reading it as much as we have enjoyed writing it. The title of this book contains two major keywords: *embedded* and *VLIW* (very long instruction word). Historically, the embedded computing community has rarely been related to the VLIW community. Technology is removing this separation, however. High-performance techniques such as VLIW that seemed too expensive for embedded designs have recently become both feasible and popular. This change is bringing in a new age of embedded computing design, in which a high-performance processor is central. More and more, the traditional elements of nonprogrammable components, peripherals, interconnects, and buses must be seen in a computing-centric light. Embedded computing designers must design systems that unify these elements with high-performance processor architectures, microarchitectures, and compilers, as well as with the compilation tools, debuggers, and simulators needed for application development.

Since this is a book about embedded computing, we define and explore that world in general, but with the strongest emphasis on the processing aspects. Then, within this new world of embedded, we show how the VLIW design philosophy matches the goals and constraints well. We hope we have done this in a way that clearly and systematically explains the unique problems in the embedded domain, while remaining approachable to those with a general background in architecture and compilation. Conversely, we also need to explain the VLIW approach and its implications and to point out the ways in which VLIW, as contrasted with other high-performance architectural techniques, is uniquely suited to the embedded world.

We think this book fills a hole in the current literature. A number of current and upcoming books cover embedded computing, but few of them take the combined hardware–software systems approach we do. While the embedded computing and digital signal processing (DSP) worlds seem exotic to those with general-purpose backgrounds, they remain *computing*. Much is common between general-purpose and embedded techniques, and after showing what is common between them, we can focus on the differences. In addition, there is no standard reference on the VLIW approach. Such a book has been needed for at least a decade, and we believe that a book explaining the VLIW design philosophy has value today. This book should be useful to engineers and designers in industry, as well as suitable as a textbook for courses that aim at seniors or first-year graduate students.

While considering the mission of our book, we came up with three different possible books on the spectrum from VLIW to embedded. The first is the previously mentioned book, purely about VLIW. The second is a book about high-performance approaches

Preface

Welcome to our book. We hope you enjoy reading it as much as we have enjoyed writing it. The title of this book contains two major keywords: *embedded* and *VLIW* (very long instruction word). Historically, the embedded computing community has rarely been related to the VLIW community. Technology is removing this separation, however. High-performance techniques such as VLIW that seemed too expensive for embedded designs have recently become both feasible and popular. This change is bringing in a new age of embedded computing design, in which a high-performance processor is central. More and more, the traditional elements of nonprogrammable components, peripherals, interconnects, and buses must be seen in a computing-centric light. Embedded computing designers must design systems that unify these elements with high-performance processor architectures, microarchitectures, and compilers, as well as with the compilation tools, debuggers, and simulators needed for application development.

Since this is a book about embedded computing, we define and explore that world in general, but with the strongest emphasis on the processing aspects. Then, within this new world of embedded, we show how the VLIW design philosophy matches the goals and constraints well. We hope we have done this in a way that clearly and systematically explains the unique problems in the embedded domain, while remaining approachable to those with a general background in architecture and compilation. Conversely, we also need to explain the VLIW approach and its implications and to point out the ways in which VLIW, as contrasted with other high-performance architectural techniques, is uniquely suited to the embedded world.

We think this book fills a hole in the current literature. A number of current and upcoming books cover embedded computing, but few of them take the combined hardware–software systems approach we do. While the embedded computing and digital signal processing (DSP) worlds seem exotic to those with general-purpose backgrounds, they remain *computing*. Much is common between general-purpose and embedded techniques, and after showing what is common between them, we can focus on the differences. In addition, there is no standard reference on the VLIW approach. Such a book has been needed for at least a decade, and we believe that a book explaining the VLIW design philosophy has value today. This book should be useful to engineers and designers in industry, as well as suitable as a textbook for courses that aim at seniors or first-year graduate students.

While considering the mission of our book, we came up with three different possible books on the spectrum from VLIW to embedded. The first is the previously mentioned book, purely about VLIW. The second is a book about high-performance approaches to the embedded domain, with equal emphasis on VLIW, Superscalar, digital signal processor (DSP), micro-SIMD (Single Instruction Multiple Data), and vector techniques. Our book (the third option) strikes a balance: it focuses on the VLIW approach to the embedded domain. This means we give lighter treatment to the alternative approaches but spend additional effort on drawing the connections between VLIW and embedded. However, large parts of the information in our book overlap material that would go into the other two, and we think of this book as valuable for those with a strong interest in embedded computing but only a little interest in VLIW, and vice versa.

Along the way, we have tried to present our particularly idiosyncratic views of embedded, VLIW, and other high-performance architectural techniques. Most of the time, we hope we have impartially presented facts. However, these topics would be terribly dry and boring if we removed all controversy. VLIW has become a significant force in embedded processing and, as we make clear, there are technical and marketing reasons for this trend to continue. We will wear our biases on our sleeves (if you can't tell from the title, we think VLIW is the correct hammer for the embedded nail), but we hope to be honest about these biases in areas that remain unresolved.

Content and Structure

When we first wrote the outline for this book, the chapters fell into three major categories: *hardware*, *software*, and *applications*. Thus, the outline of the book correspondingly had three major parts. As we have written and rewritten, the organization has changed, pieces have migrated from one chapter to another, and the clean three-part organization has broken down into a set of chapters that only roughly matches the original tripartite structure. The unfortunate truth of modern computer architecture is that one cannot consider any of hardware, software, or applications by themselves.

This book really has two introductory chapters. Chapter 1 describes the world of embedded processing. It defines embedded processing, provides examples of the various types of embedded processors, describes application domains in which embedded cores are deployed, draws distinctions between the embedded and general-purpose domains, and talks about the marketplace for embedded devices. The second introductory chapter, Chapter 2, defines *instruction-level parallelism* (ILP), the primary technique for extracting performance in many modern architectural styles, and describes how compilation is crucial to any ILP-oriented processor design. Chapter 2 also describes the notion of an architectural style or design philosophy, of which VLIW is one example. Last, Chapter 2 describes how technology has evolved so that VLIW and embedded, once vastly separate domains, are now quite suited to each other.

Chapters 3 through 5 constitute the purely "hardware"-related part of the book. Chapter 3 describes what we mean when we say architecture or instruction-set architecture (ISA), defines what a VLIW ISA looks like, and describes in particular how VLIW architectures have been built for embedded applications. Chapter 3 also describes instruction set *encoding* at two levels. From a high-level perspective, Chapter 3 revisits the notion of design philosophy and architectural style with respect to how that style affects the way operations and instructions are encoded under each design philosophy.

At a detailed level, Chapter 3 describes the particular issues associated with VLIW operation and instruction encoding.

Chapter 4 might be seen as a continuation of the previous chapter, but instead of describing ISA design as a whole (with a view across various ISA styles), Chapter 4 examines the hardware structures (such as the datapath, memory, register files, and control units) necessary to all modern processors. Chapter 4 pays particular attention to how these structures differ in the embedded domain from their general-purpose counterparts.

The next chapter explores microarchitecture, the implementation of techniques within a given ISA. Chapter 5 can be seen as largely paralleling Chapter 4 in subject matter, but it considers how to *implement* each piece of functionality rather than how to *specify* that work be done within an ISA. Chapter 5 is informed by the technological constraints of modern design; that is, wires are expensive, whereas transistors are cheap. The chapter also (very briefly) considers power-related technological concerns.

Chapter 6 fits poorly into either the hardware and software categories, as both topics occur in each of its sections. Chapter 6 begins with a description of how a system-on-a-chip (SoC) is designed. Most modern embedded systems today are designed using the SoC methodology. Chapter 6 continues with how processor cores integrate with SoCs. Then it describes simulation methodologies for processor cores, followed by simulation techniques for entire systems. Last, Chapter 6 describes validation and verification of simulators and their systems. It might be best to view Chapter 6 as a bridge between the hardware and software areas, or perhaps its integration of the two serves as a good illustration of the complexities involved in building hardware/software *systems*.

The next three chapters emphasize the software area, although reading them will make it clear that they are infused with hardware-related topics in a number of ways. Chapter 7 describes the entire *toolchain*: the suite of software programs used to analyze, design, and build the software of an embedded system. Chapter 7 also describes a number of embedded- and DSP-specific code transformations.

Chapter 8 describes a subset of the compiler optimizations and transformations in an industrial-strength ILP-oriented compiler. This book is not a compiler textbook. Our goal in this chapter is to paint a balanced picture of the suite of optimizations — including their uses, complexities, and interactions — so that system designers will understand the nature of compilation-related issues, and so that compiler designers will know where else to look.

Chapter 9 covers a broad range of topics that often fall between the cracks of traditional topics, but are nonetheless important to building a working system. Chapter 9 details issues about exceptions, application binary interfaces (ABIs), code compression, operating systems (including embedded and real-time variants), and multiprocessing. Many of these topics have a strong software component to them, but each also interacts strongly with hardware structures that support the software functionality.

The last two chapters focus on applications. Chapter 10 begins by discussing programming languages for embedded applications, and then moves on to performance, benchmarks, and tuning. Then it continues to scalability and customizability in embedded architectures, and finishes with detail about customizable processors.

Chapter 11 visits a number of embedded applications at a variety of levels of detail. We spend the most time on digital printing and imaging, and telecommunications, and less time on other areas, such as automotive, network processing, and disk drives.

While writing this book, it became clear that there are a large number of terms with overlapping and conflicting meanings in this field. For example, *instruction* can mean operation, bundle, parallel issue group, or parallel execution group to different subcommunities. Wherever possible, we use the terms as they are used in the architecture field's dominant textbook, John Hennessy and Dave Patterson's *Computer Architecture: A Quantitative Approach.* The Glossary lists alternate definitions and synonyms, and indicates which terms we intend to use consistently.

The VEX (VLIW Example) Computing System

Lest we be accused of writing an armchair textbook (like those scientists of the nineteenth century who deduced everything from first principles), our book ships with an embedded-oriented VLIW development system. We call this system VEX, for "VLIW Example." We hope it is even more useful to our readers than its textbook ancestors, MIX and DLX, were for their readers. VEX is based on production tools used at HP Labs and other laboratories. It is a piece of real-world VLIW processor technology, albeit simplified for instructional use.

VEX is intended for experimental use. It includes a number of simulators, and its tools allow hardware reconfiguration and both manual and automated design-space exploration. Code, documentation, and samples can be downloaded from the book's Web site at *http://www.vliw.org/book.* VEX examples and exercises occur throughout the book. The Appendix describes the VEX instruction set architecture and tool chain.

Audience

We assume a basic knowledge of computer architecture concepts, as might be given by some industrial experience or a first undergraduate course in architecture. This implies that you know the basic techniques of pipelining and caching, and that the idea of an instruction set is familiar. It helps but is not a requirement that you have some background in compilation, or at least that you believe an optimizing compiler might be useful in producing fast code for modern machines. For reasons of space, we touch on those fundamentals related to this text and for more basic information refer you to more basic architecture and compilation textbooks. Patterson and Hennessy's undergraduate architecture textbook, *Computer Organization and Design*, and Appel's polymorphic set of undergraduate compiler books, *Modern Compiler Implementation in C, Java*, and *ML* are fine places to start.

There are four likely types of readers of our book. For those trying to bridge the embedded and high-performance communities, we believe this book will help. Designers of general-purpose systems interested in embedded issues should find this book a useful introduction to a new area. Conversely, those who work with existing embedded and/or DSP designs but would like to understand more about high-performance computing in