Architecture Assembly Language Programming Edition

Assembly is a low-level programming language that's one step above a computer's native machine language. Although assembly language is commonly used for writing device drivers, emulators, and video games, many programmers find its somewhat unfriendly syntax intimidating to learn and use. Since 1996, Randall Hyde's The Art of Assembly Language has provided a comprehensive, plain-English, and patient introduction to 32-bit x86 assembly for non-assembly programmers. Hyde's primary teaching tool, High Level Assembler (or HLA), incorporates many of the features found in high-level languages (like C, C++, and Java) to help you quickly grasp basic assembly concepts. HLA lets you write true low-level code while enjoying the benefits of high-level language programming. As you read The Art of Assembly Language, you'll learn the low-level theory fundamental to computer science and turn that understanding into real, functional code. You'll learn how to: -Edit, compile, and run HLA programs -Declare and use constants, scalar variables, pointers, arrays, structures, unions, and namespaces -Translate arithmetic expressions (integer and floating point) -Convert high-level control structures This much anticipated second edition of The Art of Assembly Language has been updated to reflect recent changes to HLA and to support Linux, Mac OS X, and FreeBSD. Whether you're new to programming or you have experience with high-level languages, The Art of Assembly Language, 2nd Edition is your essential guide to learning this complex, low-level language.

The predominant language used in embedded microprocessors, assembly language lets you write programs that are typically faster and more compact than programs written in a high-level language and provide greater control over the program applications. Focusing on the languages used in X86 microprocessors, X86 Assembly Language and C Fundamentals explains how to write programs in the X86 assembly language, the C programming language, and X86 assembly language modules embedded in a C program. A wealth of program design examples, including the complete code and outputs, help you grasp the concepts more easily. Where needed, the book also details the theory behind the design. Learn the X86 Microprocessor Architecture and Commonly Used Instructions Assembly language programming requires knowledge of number representations, as well as the architecture of the computer on which the language is being used. After covering the binary, octal, decimal, and hexadecimal number systems, the book presents the general architecture of the X86 microprocessor, individual addressing modes, stack operations, procedures, arrays, macros, and input/output operations. It highlights the most commonly used X86 assembly language instructions, including data transfer, branching and looping, logic, shift and rotate, and string instructions, as well as fixed-point, binary-coded decimal (BCD), and floating-point arithmetic instructions. Get a Solid Foundation in a Language Commonly Used in Digital Hardware Written for students in computer science and electrical, computer, and software engineering, the book assumes a basic background in C programming, digital logic design, and computer architecture. Designed as a tutorial, this comprehensive and self-contained text offers a solid foundation in assembly language for anyone working with the design of digital hardware.

The performance of software systems is dramatically affected by how well software designers understand the basic hardware technologies at work in a system. Similarly, hardware designers must understand the far-reaching effects their design decisions have on software applications. For readers in either category, this classic introduction to the field provides a look deep into the computer. It demonstrates the relationships between the software and hardware and focuses on the foundational concepts that are the basis for current computer design.

Mastering ARM hardware architecture opens a world of programming for nearly all phones and tablets including the iPhone/iPad and most Android phones. It’s also the heart of many single board computers like the Raspberry Pi. Gain the skills required to dive into the fundamentals of the ARM hardware architecture with this book and start your own projects while you develop a working knowledge of assembly language for the ARM 64-bit processor. You'll review assembly language programming for the ARM Processor in 64-bit mode and write programs for a number of single board computers, including the Nvidia Jetson Nano and the Raspberry Pi (running 64-bit Linux). The book also discusses how to target assembly language programs for Apple iPhones and iPads along with 64-Bit ARM based Android phones and tablets. It covers all the tools you require, the basics of the ARM hardware architecture, all the groups of ARM 64-Bit Assembly instructions, and how data is stored in the computer’s memory. In addition, interface apps to hardware such as the Raspberry Pi’s GPIO ports. The book covers code optimization, as well as how to inter-operate with C and Python code. Readers will develop enough background to use the official ARM reference documentation for their own projects. With Programming with 64-Bit ARM Assembly Language as your guide you’ll study how to read, reverse engineer and hack machine code, then be able to apply these new skills to study code examples and take control of both your ARM devices’ hardware and software. What You'll LearnMake operating system calls from assembly language and include other software libraries in your projects Interface apps to hardware devices such as the Raspberry Pi GPIO ports Reverse engineer and hack code Use the official ARM reference documentation for your own projects Who This Book Is For Software developers who have already learned to program in a higher-level language like Python, Java, C#, or even C and now wish to learn Assembly programming.

The Essentials of Computer Organization and Architecture
Low-Level Programming
Assembly Language Programming for Intel Processors Family
Fundamentals and Techniques, Second Edition
VAX
Structured Assembly Language Programming

Takes a unique systems approach to programming and architecture of the VAX Using the VAX as a detailed example, the first half of this book offers a complete course in assembly language programming. The second describes higher-level systems issues in computer architecture. Highlights include the VAX assembler and debugger, other modern architectures such as RISCs, multiprocessing and parallel computing, microprogramming, caches and translation buffers, and an appendix on the Berkeley UNIX assembler.

Users of this book will gain an understanding of the fundamental concepts of contemporary computer architecture, starting with a Reduced Instruction Set Computer (RISC). An understanding of computer architecture needs to begin with the basics of modern computer organization. The MIPS architecture embodies the fundamental design principles of all contemporary RISC architectures. This book provides an understanding of how the functional components of modern computers are put together and how a computer works at the machine-language level. Well-written and clearly organized, this book covers the basics of MIPS architecture, including algorithm development, number systems, function calls, reentrant functions, memory-mapped I/O, exceptions and interrupts, and floating-point instructions. For employees in the field of systems, systems development, systems analysis, and systems maintenance.

This is a straightforward text on RISC assembly language programming for MIPS computers - the microprocessor gaining popularity due to its compact and elegant instruction set. Enabling students to understand the internal working of a computer, courses in RISC are an increasingly popular option in assembly language programming.

This updated textbook introduces readers to assembly and its evolving role in computer programming and design. The author concentrates the revised edition on protected-mode Pentium programming, MIPS assembly language programming, and use of the NASM and SPIM assemblers for a Linux orientation. The focus is on providing students with a firm grasp of the main features of assembly programming, and how it can be used to improve a computer's performance. All of the main features are covered in depth, and the book is equally viable for DOS or Linux, MIPS (RISC) or CISC (Pentium). The book is based on a successful course given by the author and includes numerous hands-on exercises.

A Concise Introduction
Arm Assembly Language - An Introduction (Second Edition)
Covers x86 64-bit, AVX, AVX2, and AVX-512
The Vax
An Introduction to Professional C Programming
Assembly Language for X86 Processors

Gain all the skills required to dive into the fundamentals of the Raspberry Pi hardware architecture and how data is stored in the Pi’s memory. This book provides you with working starting points for your own projects while you develop a working knowledge of Assembly language progra on the Raspberry Pi. You'll learn how to interface to the Pi’s hardware including accessing the GPIO ports. The book will cover the basics of code optimization as well as how to inter-operate with C and Python code, so you'll develop enough background to use the official ARM reference documentation for further projects. With Raspberry Pi Assembly Language Programming as your guide you'll study how to read and reverse engineer machine code and then then apply those new skills to study code examples and take control of your Pi’s hardware and software both. W Learn Program basic ARM 32-Bit Assembly Language Interface with the various hardware devices on the Raspberry Pi Comprehend code containing Assembly language Use the official ARM reference documentation Who This Book Is For Coders who have already learned to program higher-level language like Python, Java, C#, or C and now wish to learn Assembly programming.

The increasing complexity of programming environments provides a number of opportunities for assembly language programmers. 32/64-Bit 80x86 Assembly Language Architecture attempts to break through that complexity by providing a step-by-step understanding of programming Int AMD 80x86 processors in assembly language. This book explains 32-bit and 64-bit 80x86 assembly language programming inclusive of the SIMD (single instruction multiple data) instruction supersets that bring the 80x86 processor into the realm of the supercomputer, gives insight into (floating-point unit) chip in every Pentium processor, and offers strategies for optimizing code.

"Essentials of 80x86 Assembly Language" is designed as a supplemental text for the instructor who wants to provide students hands-on experience with the Intel 80x86 architecture. It can also be used as a stand-alone text for an assembly language course.

A detailed introduction to the C programming language for experienced programmers. The world runs on code written in the C programming language, yet most schools begin the curriculum with Python or Java. Effective C bridges this gap and brings C into the modern era--covering the C17 Standard as well as potential C2x features. With the aid of this instant classic, you'll soon be writing professional, portable, and secure C programs to power robust systems and solve real-world problems. Robert C. Seacord introduces C and the C Standard Library while addressing practices, common errors, and open debates in the C community. Developed together with other C Standards committee experts, Effective C will teach you how to debug, test, and analyze C programs. You'll benefit from Seacord's concise explanations of C language constructs and be from his 40 years of coding experience. You'll learn: • How to identify and handle undefined behavior in a C program • The range and representations of integers and floating-point values • How dynamic memory allocation works and how to use nonstandard functions • How to use chara encodings and types • How to perform I/O with terminals and filesystems using C Standard streams and POSIX file descriptors • How to understand the C compiler's translation phases and the role of the preprocessor • How to test, debug, and analyze C programs Effective C will teach write professional, secure, and portable C code that will stand the test of time and help strengthen the foundation of the computing world.

Computer Architecture and VAX Assembly Language Programming
An Introduction to Assembly Language Programming and Computer Architecture
Computer Organization and Design
ARM Assembly Language
32/64-Bit 80x86 Assembly Language Architecture
Assembly Language Programming

Learn Intel 64 assembly language and architecture, become proficient in C, and understand how the programs are compiled and executed down to machine instructions, enabling you to write robust, high-performance code. Low-Level Programming explains Intel 64 architecture as the result of von Neumann architecture evolution. The book teaches the latest version of the C language (C11) and assembly language from scratch. It covers the entire path from source code to program execution, including generation of ELF object files, and static and dynamic linking. Code examples and exercises are included along with the best code practices. Optimization capabilities and limits of modern compilers are examined, enabling you to balance between program readability and performance. The use of various performance-gain techniques is demonstrated, such as SSE instructions and pre-fetching. Relevant Computer Science topics such as models of computation and formal grammars are addressed, and their practical value explained. What You'll Learn Low-Level Programming teaches programmers to: Freely write in assembly language Understand the programming model of Intel 64 Write maintainable and robust code in C11 Follow the compilation process and decipher assembly listings Debug errors in compiled assembly code Use appropriate models of computation to greatly reduce program complexity Write performance-critical code Comprehend the impact of a weak memory model in multi-threaded applications Who This Book Is For Intermediate to advanced programmers and programming students

Intelligent readers who want to build their own embedded computer systems-- installed in everything from cell phones to cars to handheld organizers to refrigerators-- will find this book to be the most in-depth, practical, and up-to-date guide on the market. Designing Embedded Hardware carefully steers between the practical and philosophical aspects, so developers can both create their own devices and gadgets and customize and extend off-the-shelf systems. There are hundreds of books to choose from if you need to learn programming, but only a few are available if you want to learn to create hardware. Designing Embedded Hardware provides software and hardware engineers with no prior experience in embedded systems with the necessary conceptual and design building blocks to understand the architectures of embedded systems. Written to provide the depth of coverage and real-world examples developers need, Designing Embedded Hardware also provides a road-map to the pitfalls and traps to avoid in designing embedded systems. Designing Embedded Hardware covers such essential topics as: The principles of developing computer hardware Core hardware designs Assembly language concepts Parallel I/O Analog-digital conversion Timers (internal and external) UART Serial Peripheral Interface Inter-Integrated Circuit Bus Controller Area Network (CAN) Data Converter Interface (DCI) Low-power operation This invaluable and eminently useful book gives you the practical tools and skills to develop, build, and program your own application-specific computers.

Assembly language is as close to writing machine code as you can get without writing in pure hexadecimal. Since it is such a low-level language, it's not practical in all cases, but should definitely be considered when you're looking to maximize performance. With Assembly Language by Chris Rose, you'll learn how to write x64 assembly for modern CPUs, first by writing inline assembly for 32-bit applications, and then writing native assembly for C++ projects. You'll learn the basics of memory spaces, data segments, CISC instructions, SIMD instructions, and much more. Whether you're working with Intel, AMD, or VIA CPUs, you'll find this book a valuable starting point since many of the instructions are shared between processors.This updated and expanded second edition of Book provides a user-friendly introduction to the subject, Taking a clear structural framework, it guides the reader through the subject's core elements. A flowing writing style combines with the use of illustrations and diagrams throughout the text to ensure the reader understands even the most complex of concepts. This succinct and enlightening overview is a required reading for all those interested in the subject .We hope you find this book useful in shaping your future career & Business.

An assembly (or assembler) language, often abbreviated asm, is a low-level programming language for a computer, or other programmable device, in which there is a very strong (generally one-to-one) correspondence between the language and the architecture's machine code instructions. Each assembly language is specific to a particular computer architecture. In contrast, most high-level programming languages are generally portable across multiple architectures but require interpreting or compiling. Assembly language may also be called symbolic machine code. Assembly language is converted into executable machine code by a utility program referred to as an assembler. The conversion process is referred to as assembly, or assembling the source code. Assembly time is the computational step where an assembler is run. This updated and expanded second edition of Book provides a user-friendly introduction to the subject, Taking a clear structural framework, it guides the reader through the subject's core elements. A flowing writing style combines with the use of illustrations and diagrams throughout the text to ensure the reader understands even the most complex of concepts. This succinct and enlightening overview is a required reading for all those interested in the subject . We hope you find this book useful in shaping your future career & Business.

Assembly Language for Students
With Assembly Language Examples from the MIPS RISC Architecture
The Art of Assembly Language, 2nd Edition
ARM Processor Coding
Introduction to RISC Assembly Language Programming
Arm Assembly Language Programming & Architecture

ARM designs the cores of microcontrollers which equip most "embedded systems" based on 32-bit processors. Cortex M3 is one of these designs, recently developed by ARM with microcontroller applications in mind. To conceive a particularly optimized piece of software (as is often the case in the world of embedded systems) it is often necessary to know how to program in an assembly language. This book explains the basics of programming in an assembly language, while being based on the architecture of Cortex M3 in detail and developing many examples. It is written for people who have never programmed in an assembly language and is thus didactic and progresses step by step by defining the concepts necessary to acquiring a good understanding of these techniques.

An introductory text describing the ARM assembly language and its use for simple programming tasks.

This book is about two separate but related topics: assembly language programming and computer architecture. This is based on the notion that it is not possible to study computer architecture in any depth without some knowledge of assembly language programming and similarly, one of the reasons for studying assembly language programming is to gain an insight into how computers work - which naturally leads to their architecture. Introducing Assembly Language Programming and Computer Architecture is ideal for first year computer science or engineering students taking degree and diploma level courses. It will also be a useful reference for computer enthusiasts wishing to advance their knowledge and programming skills.

This introductory text offers a contemporary treatment of computer architecture using assembly and machine language with a focus on software. Students learn how computers work through a clear, generic presentation of a computer architecture, a departure from the traditional focus on a specific architecture. A computer's capabilities are introduced within the context of software, reinforcing the software focus of the text. Designed for computer science majors in an assembly language course, this text uses a top-down approach to the material that enables students to begin programming immediately and to understand the assembly language, the interface between hardware and software. The text includes examples from the MIPS RISC (reduced instruction set computer) architecture, and an accompanying software simulator package simulates a MIPS RISC processor (the software does not require a MIPS processor to run).

C, Assembly, and Program Execution on Intel® 64 Architecture
Third Edition
Assembly Programming and Computer Architecture
Computer Organization and Assembly Language Programming
Introduction to 80x86 Assembly Language and Computer Architecture
Assembly Language Step-by-Step

A Revised and Updated Edition of the Authoritative Text This revised and updated Third Edition of the classic text guides students through assembly language using a hands-on approach, supporting future computing professionals with the basics they need to understand the mechanics and function of the computer's inner workings. Through using real instruction sets to write real assembly language programs, students will become acquainted with the basics of computer architecture. 80x86 Assembly Language and Computer Architecture covers the Intel 80x86 using the powerful tools provided by Microsoft Visual Studio, including its 32- and 64-bit assemblers, its versatile debugger, and its ability to link assembly language and C/C++ program segments. The text also includes multiple examples of how individual 80x86 instructions execute, as well as complete programs using these instructions. Hands-on exercises reinforce key concepts and problem-solving skills. Updated to be compatible with Visual Studio 2012, and incorporating over a hundred new exercises, 80x86 Assembly Language and Computer Architecture: Third Edition is accessible and clear enough for beginning students while providing coverage of a rich set of 80x86 instructions and their use in simple assembly language programs. The text will prepare students to program effectively at any level. Key features of the fully revised and updated Third Edition include: ▯ Updated to be used with Visual Studio 2012, while remaining compatible with earlier versions ▯ Over 100 new exercises and programming exercises ▯ Improved, clearer layout with easy-to-read illustrations ▯ The same clear and accessibly writing style as previous editions ▯ Full suite of ancillary materials, including PowerPoint lecture outlines, Test Bank, and answer keys ▯ Suitable as a stand-alone text in an assembly language course or as a supplement in a computer architecture course

Detailed coverage of architecture/hardware topics such as CPU, microprocessors, large computer architecture and fault tolerance architecture makes this a valuable reference. For computer science and electrical engineering professionals as well as VAX assembly language programmers.

The eagerly anticipated new edition of the bestselling introduction to x86 assembly language The long-awaited third edition of this bestselling introduction to assembly language has been completely rewritten to focus on 32-bit protected-mode Linux and the free NASM assembler. Assembly is the fundamental language bridging human ideas and the pure silicon hearts of computers, and popular author Jeff Dunteman retains his distinctive lighthearted style as he presents a step-by-step approach to this difficult technical discipline. He starts at the very beginning, explaining the basic ideas of programmable computing, the binary and hexadecimal number systems, the Intel x86 computer architecture, and the process of software development under Linux. From that foundation he systematically treats the x86 instruction set, memory addressing, procedures, macros, and interface to the C-language code libraries upon which Linux itself is built. Serves as an ideal introduction to x86 computing concepts, as demonstrated by the only language directly understood by the CPU itself Uses an approachable, conversational style that assumes no prior experience in programming of any kind Presents x86 architecture and assembly concepts through a cumulative tutorial approach that is ideal for self-paced instruction Focuses entirely on free, open-source software, including Ubuntu Linux, the NASM assembler, the Kate editor, and the

Gdb/Insight debugger Includes an x86 instruction set reference for the most common machine instructions, specifically tailored for use by programming beginners Woven into the presentation are plenty of assembly code examples, plus practical tips on software design, coding, testing, and debugging, all using free, open-source software that may be downloaded without charge from the Internet.
Structured VAX Assembly Language Programming, Second Edition, provides a complete, up-to-date introduction to VAX programming and the fundamentals of VAX architecture. The book emphasizes sound, structured programming techniques that are modelled in a number of new program examples. The text also features complete chapters on RMS, and the VAX VMS-debugger, including a new discussion of using the debugger in the screen mode. This is a comprehensive, well-organized text and reference for both students and professional programmers.Features * A complete chapter on RMS including the VMS sub-system used in high-level VAX languages for input and output. * Expanded chapter on the VAX-VMS debugger that shows how to use commands efficiently to moniter program execution, and how to use the debugger in screen mode. * Expanded coverage of VAX architecture fundamentals. * A structured approach to assembly language programming that reinforces structured programming concepts. * Many new program examples. This site also contains the two macro files formerly available at ftp: //happy.uccs.colorado.edu/macro. That site no longer exists, so the macros have been moved here: iomac.mar iosub.mar 0805371222B04062
Computer Programming and Architecture
A Programmer's View of Computer Architecture
Modern X86 Assembly Language Programming
32-bit, 64-bit, SSE, and AVX

MIPS Assembly Language Programming
For Assembly Language and Architecture courses emphasizing SPARC architecture found in computer science, engineering and business departments. Written from a programmer's perspective, this long-awaited revision introduces the SPARC assembly language to readers early on. Other introductory material encompasses making use of UNIX tools (the m4 macro processor; the assembler; the gnu emacs editor; and the gdb debugger). Further coverage includes a formal definition of the von Neumann machine, its relationship to programmable calculators, and to the JAVA bytecode and JAVA virtual machine. Not only is this book suitable for introductory computer architecture courses, but for programmers who will be programming SPARC architecture machine in languages such as C and C++.
Computer Architecture/Software Engineering
Modern X86 Assembly Language Programming shows the fundamentals of x86 assembly language programming. It focuses on the aspects of the x86 instruction set that are most relevant to application software development. The book's structure and sample code are designed to help the reader quickly understand x86 assembly language programming and the computational capabilities of the x86 platform. Please note: Book appendixes can be downloaded here: http://www.apress.com/9781484200650 Major topics of the book include the following: 32-bit core architecture, data types, internal registers, memory addressing modes, and the basic instruction set X87 core architecture, register stack, special purpose registers, floating-point encodings, and instruction set MMX technology and instruction set Streaming SIMD extensions (SSE) and Advanced Vector Extensions (AVX) including internal registers, packed integer arithmetic, packed and scalar floating-point arithmetic, and associated instruction sets 64-bit core architecture, data types, internal registers, memory addressing modes, and the basic instruction set 64-bit extensions to SSE and AVX technologies X86 assembly language optimization strategies and techniques
Delivering a solid introduction to assembly language and embedded systems, ARM Assembly Language: Fundamentals and Techniques, Second Edition continues to support the popular ARM7TDMI, but also addresses the latest architectures from ARM, including CortexTM-A, Cortex-R, and Cortex-M processors—all of which have slightly different instruction sets, programmer ' s models, and exception handling. Featuring three brand-new chapters, a new appendix, and expanded coverage of the ARM7TM, this edition: Discusses IEEE 754 floating-point arithmetic and explains how to program with the IEEE standard notation Contains step-by-step directions for the use of KeilTM MDK-ARM and Texas Instruments (TI) Code Composer StudioTM Provides a resource to be used alongside a variety of hardware evaluation modules, such as TI ' s Tiva Launchpad, STMicroelectronics ' iNemo and Discovery, and NXP Semiconductors ' Xplorer boards Written by experienced ARM processor designers, ARM Assembly Language: Fundamentals and Techniques, Second Edition covers the topics essential to writing meaningful assembly programs, making it an ideal textbook and professional reference.
An Introduction to Intel Assembly Language
For Software Engineers
Introduction to 80 X 86 Assembly Language and Computer Architecture
For Pentium and RISC Processors
Single Board Computer Development for Raspberry Pi and Mobile Devices
Designing Embedded Hardware
Gain the fundamentals of x86 64-bit assembly language programming and focus on the updated aspects of the x86 instruction set that are most relevant to application software development. This book covers topics including x86 64-bit programming and Advanced Vector Extensions (AVX) programming. The focus in this second edition is exclusively on 64-bit base programming architecture and AVX programming. Modern X86 Assembly Language Programming's structure and sample code are designed to help you quickly understand x86 assembly language programming and the computational capabilities of the x86 platform. After reading and using this book, you'll be able to code performance-enhancing functions and algorithms using x86 64-bit assembly language and the AVX, AVX2 and AVX-512 instruction set extensions. What You Will Learn Discover details of the x86 64-bit platform including its core architecture, data types, registers, memory addressing modes, and the basic instruction set Use the x86 64-bit instruction set to create performance-enhancing functions that are callable from a high-level language (C++) Employ x86 64-bit assembly language to efficiently manipulate common data types and programming constructs including integers, text strings, arrays, and structures Use the AVX instruction set to perform scalar floating-point arithmetic Exploit the AVX, AVX2, and AVX-512 instruction sets to significantly accelerate the performance of computationally-intense algorithms in problem domains such as image processing, computer graphics, mathematics, and statistics Apply various coding strategies and techniques to optimally exploit the x86 64-bit, AVX, AVX2, and AVX-512 instruction sets for maximum possible performance Who This Book Is For Software developers who want to learn how to write code using x86 64-bit assembly language. It's also ideal for software developers who already have a basic understanding of x86 32-bit or 64-bit assembly language programming and are interested in learning how to exploit the SIMD capabilities of AVX, AVX2 and AVX-512.
Modern Assembly Language Programming with the ARM Processor is a tutorial-based book on assembly language programming using the ARM processor. It presents the concepts of assembly language programming in different ways, slowly building from simple examples towards complex programming on bare-metal embedded systems. The ARM processor was chosen as it has fewer instructions and irregular addressing rules to learn than most other architectures, allowing more time to spend on teaching assembly language programming concepts and good programming practice. In this textbook, careful consideration is given to topics that students struggle to grasp, such as registers vs. memory and the relationship between pointers and addresses, recursion, and non-integral binary mathematics. A whole chapter is dedicated to structured programming principles. Concepts are illustrated and reinforced with a large number of tested and debugged assembly and C source listings. The book also covers advanced topics such as fixed and floating point mathematics, optimization, and the ARM VFP and NEONTM extensions. PowerPoint slides and a solutions manual are included. This book will appeal to professional embedded systems engineers, as well as computer engineering students taking a course in assembly language using the ARM processor. Concepts are illustrated and reinforced with a large number of tested and debugged assembly and C source listing Intended for use on very low-cost platforms, such as the Raspberry Pi or pcDuino, but with the support of a full Linux operating system and development tools Includes discussions of advanced topics, such as fixed and floating point mathematics, optimization, and the ARM VFP and NEON extensions
SPARC Architecture, Assembly Language Programming, and CPearson
Clements has a gift for conveying highly complex, technical information in an exceptionally clear and readable manner. Clements writing style is very student oriented, and stresses the basics of 68000 ASL while also covering the latest information on ASL later generation chips.
Assembly Language and Computer Architecture Using C++ and Java
68000 Family Assembly Language
Programming with 64-Bit ARM Assembly Language
Introduction to Assembly Language Programming
Modern Assembly Language Programming with the ARM Processor
SPARC Architecture, Assembly Language Programming, and C
The objective of this book is to make it possible (and even easy) for students to master both assembly language and the fundamentals of computer architecture in a single semester. Integrating coverage of software and hardware throughout, the book uses H1--a simple, horizontally microprogrammed computer--as a unifying theme. Like all simple models, H1 has flaws, but this book puts these flaws to good use. In particular, in addition to showing students how H1 works and what is wrong with it, the book shows students how to fix it (which they then proceed to do). Students learn best by doing, and this book supplies much to do with various examples and projects to facilitate learning. For example, students not only use assemblers and linkers, they also write their own. Students not only study and use the provided instruction set but implement new, improved ones. The result is a book that is easy to read, engaging, and substantial. The software package for the book supports Windows, Mac OS X, Linux, and Raspbian.
Who uses ARM? Currently ARM CPU is licensed and produced by more than 200 companies and is the dominant CPU chip in both cell phones and tablets. Given its RISC architecture and powerful 32-bit instructions set, it can be used for both 8-bit and 32-bit embedded products. The ARM corp. has already defined the 64-bit instruction extension and for that reason many Laptop and Server manufactures are introducing ARM-based Laptop and Servers. Who will use our textbook? This book is intended for both academic and industry readers. If you are using this book for a university course, the support materials and tutorials can be found on www.MicroDigitalEd.com. This book covers the Assembly language programming of the ARM chip. The ARM Assembly language is standard regardless of who makes the chip. The ARM licensees are free to implement the on-chip peripheral (ADC, Timers, I/O, etc.) as they choose. Since the ARM peripherals are not standard among the various vendors, we have dedicated a separate book to each vendor.
This concise guide is designed to enable the reader to learn how to program in assembly language as quickly as possible. Through a hands-on programming approach, readers will also learn about the architecture of the Intel processor, and the relationship between high-level and low-level languages. This updated second edition has been expanded with additional exercises, and enhanced with new material on floating-point numbers and 64-bit processing. Topics and features: provides guidance on simplified register usage, simplified input/output using C-like statements, and the use of high-level control structures; describes the implementation of control structures, without the use of high-level structures, and often with related C program code; illustrates concepts with one or more complete program; presents review summaries in each chapter, together with a variety of exercises, from short-answer questions to programming assignments; covers selection and iteration structures, logic, shift, arithmetic shift, rotate, and stack instructions, procedures and macros, arrays, and strings; includes an introduction to floating-point instructions and 64-bit processing; examines machine language from a discovery perspective, introducing the principles of computer organization. A must-have resource for undergraduate students seeking to learn the fundamentals necessary to begin writing logically correct programs in a minimal amount of time, this work will serve as an ideal textbook for an assembly language course, or as a supplementary text for courses on computer organization and architecture. The presentation assumes prior knowledge of the basics of programming in a high-level language such as C, C++, or Java.
Updated and revised, The Essentials of Computer Organization and Architecture, Third Edition is a comprehensive resource that addresses all of the necessary organization and architecture topics, yet is appropriate for the one-term course.
Programming with Linux
Guide to Assembly Language
Essentials of 80x86 Assembly Language
ARM Cortex-M3
Raspberry Pi Assembly Language Programming
Effective C