Assembly Language for Intel-Based Computers, Kip R. Irvine, Prentice Hall PTR, 2003, 0130910139, 9780130910134, 708 pages. This present edition of a widely-used book provides basic information for the beginning programmer interested in computer architecture, operating system, and compiler writing. Using the Intel processor family as its base, and concentrating on the MS-DOS operating system, this complete and fully updated study of assembly language is written clearly and straightforwardly, making it easy to read and understand. A companion CD-ROM with Microsoft Macro Assembler Version 6.15 and TextPad Shareware editor is included, and there is a companion Website maintained by the author. Introduced by a chapter on Basic Concepts, the book covers machine architecture, processor architecture, assembly language fundamentals, data transfers, addressing and arithmetic, procedures, conditional processing, integer arithmetic, strings and arrays, structures and macros, 32-bit Windows programming, language interface, disk fundamentals, BIOS-level programming, and MS-DOS programming.

DOWNLOAD HERE

Assembly language programming and organization of the IBM PC, Ytha Y. Yu, Charles Marut, Dec 1, 1992, Computers, 544 pages. This introduction to the organization and programming of the 8086 family of microprocessors used in IBM microcomputers and compatibles is comprehensive and thorough. Includes ....

Assembly language programming for the 80386, Judi N. Fernández, Ruth Ashley, 1990, Computers, 388 pages. The authors provide a comprehensive guide to Assembly language for the Intel 8086 family through to 80386. The programmer reading this book will be able to write efficient ....


Java: an introduction to computer science and programming, Volume 1 an introduction to computer science and programming, Walter J. Savitch, 2001, Computers, 1049 pages. Appropriate for introductory Computer Science courses using Java (CS1 with Java) and other introductory programming courses using Java offered in Computer Science, Computer ....


Professional Assembly Language Programming, Richard Blum, Feb 1, 2007, , 576 pages. Wrox's Professional Assembly Language Programming teaches professional programmers how to incorporate assembly language programming into new and existing program projects, and ....

Introduction to assembly language programming from 8086 to Pentium processors, Sivarama P. Dandamudi, 1998, Computers, 644 pages. Introducing the reader to assembly language programming and its role in computer programming and design, this title covers stacks, addressing modes, arithmetic, selection and ....


The Waite Group's Microsoft macro assembler bible, Nabajyoti Barkakati, Randall Hyde, 1992, Computers, 952 pages. This book explores all of the new features including improved data types support, enhanced macro capabilities, single-pass operation, and a low-level optimizer. Also, any ....

Fundamentals of multimedia, Ze-Nian Li, Mark S. Drew, 2004, , 560 pages. This book offers introductory-to-advanced material on all major aspects of multimedia, including pointers to current links for information and demos at the most advanced level ....


Assembly language for real programmers only, Marcus Johnson, 1993, Computers, 1356 pages. Take advantage of the power of assembly language programming with Assembly Language: For Real Programmers ONLY! This combination tutorial and reference includes all the ....

Assembly Language for Intel-Based and Visual C++ Express 2005 CD, Kip R. Irvine, Jun 16, 2006, , 722 pages. This widely used, fully updated assembly language book provides basic information for the beginning programmer interested in computer architecture, operating systems, hardware ....

Mastering Turbo Assembler, Tom Swan, 1989, Computers, 695 pages. Mastering Turbo Assembler is a hands-on tutorial combined with an advanced-level reference, making this book an excellent resource for MS-DOS programmers using Borland's newest ....

Written specifically for the Intel/Windows/DOS platform, this complete and fully updated study of assembly language teaches students to write and debug programs at the machine level. Based on the Intel processor family, the text simplifies and demystifies concepts that students need to grasp before they can go on to more advanced computer architecture and operating systems courses. Students put theory into practice through writing software at the machine level, creating a memorable experience that gives them the confidence to work in any OS/machine-oriented environment. Proficiency in one other programming language, preferably Java, C, or C++, is recommended.

"This textbook [Irvine] teaches assembly and architecture fundamentals in a logical and concise manner for students with a reasonable CS1 backgrounds...and are applicable to higher-level programmers as to their understanding of what is happening to the code that they write and how it behaves during compilation and execution." — John Doyle, Indiana University, Southeast

"The problems and exercises are of good quality and quantity; they always have similarity within the chapter examples, but they are presented in a more challenging way. Students can pick-up skills that can be transferred to solving a new problem." — Yinping Jiao, South Texas College
The book [Irvine] is well-organized. The chapters are lined-up such that after you cover the foundations presented in chapter 1-8, you can jump to any chapter you like consistent with what you think the students should know for upcoming classes.

Wonderful! This edition added clearer examples to make it easy to understand assembly language and computer architecture from the programmer's point-of-view. I am very impressed with the clarity of explanations. The diagrams are simple and complement the discussions perfectly.

The huff about the "Irvine libraries" from other reviewers is vastly overblown out of all proportion. The POINT of the libraries is to give the new assembly language programmer some basic tools so that meaningful programs can be immediately written without getting into ALL the gory details.

Disclaimer: I have no connection to the author and had never heard of him before reading this book, but in his defense to "reviewing his own book" which he used to respond politely and while openly declaring his identity; he was required to give it SOME rating when doing that as it is part of submitting the review.

Using the "invoke" command was trivial for me from the start, and really allowed me to take advantage of both the dynamic C-runtime library (dynamic DLL, included on all Windows machines since at least Win2000) and the Win32 API from the very beginning. But again, he has a chapter (later) in the book on using that method.

You can write a LOT of very fast and EXTREMELY SMALL code by using "invoke" this way with the CRT and Win32 API. If you do this, be sure to link to the STANDARD msvcrt.lib/dll and NOT one of the VC specific versions like msvcr100.dll or msvcr70.dll until you know what the advantages and disadvantages are.

Assembly language is a tough topic, and requires thinking in different ways than you might be used to (doing math in binary and hexadecimal, and converting between these and decimal). You need to work with registers and memory in very low level ways. This book doesn't make the topic fun or easy, but it does make it very clear and learnable. The book is arranged in a logical progression of topics, and each chapter breaks the topics down in very clear and understandable ways. The material is formatted nicely for going back and referencing material. The end of chapter questions and exercises are very helpful and useful.

Writing assembly language? I give one star. But considering it is a necessary evil of a university computer science program, I accept it. The book gets 5 stars for making it tolerable. Oh - and doing math in binary and hexadecimal? I give that 5 stars! When you first see it, it looks like learning Chinese. But once you understand it, it's very logical and interesting. And then when you explain how it works to a non-techie friend, you sound like you're visiting here from outer space!

Initially, I hated this book. Hated it. But as I progressed through it I grudgingly had to admit that I was learning. Let me state unequivocally, this IS a textbook - it's priced like one and it reads like one. It's not a self study book like the Head First, Deitel, Sams, Dummies, O'Reilly, et al series, but I think with a modest amount of effort one can learn using it.

This is the 6th edition of this book. The Sixth!! The front matter states that this book was previously published as "Assembly Language for Intel-Based Computers" which itself has been around since the early 90s. There are signs that the author has made attempts to drag this book into the current millennium but it's still got more age spots than a 3 month old banana.

Another example, in chapter 5 he details opening/modifying the console window. I would think that anyone who has experience programming in C/Java/Python (as per the prereqs in 1.1.1) would already know what the console window is. This suggests that this is one of the many updates made
to this book of the last decade or so which only adds to its fractured appearance.

The bottom line is that you can only slap so much paint on an old house. It's like describing how to run a post office using a manual that was originally written for a pony express rider. I think it's time the author wrote a brand new book from the ground up. Maybe its title could be "Assembly Programming for the 21st Century".

Whereas most programming books I've used involve a lot of "doing", i.e. you're programming each step of the way, in true textbook fashion this book does more "telling". Other books will dole out info followed by exercises several times throughout a chapter. With this book, the author data dumps on you and saves the programming exercises until the end of the chapter. Then there are the author's link libraries:

That last part is easier said than done, especially if someone needs instruction on how to open a console window. But I don't know why the author made using his libraries integral to the majority of his examples. He's not alone in this by the way - at least he promotes using an industry recognized environment (Microsoft's Visual Studio). Another book, Art of Assembly, teaches assembly using the author's home-grown assembler & libraries. Anyway, if we're learning on MASM, why not use the libraries included in MASM? Especially for the basic stuff? Or better yet, use the debugger to follow your output.

"Our first version of the AddSub program used the Irvine32.inc file, which hides a few details. Eventually you will understand everything in that file, but we're just getting started in assembly language. If you prefer full disclosure of information from the start, here is a version of AddSub that does not depend on include files."

Kind of implies that you don't need the Irvine libraries, right? But at the end of the section there's this "...DumpRegs is a procedure from the Irvine32 link library that displays registers." So this example code will absolutely not work without the Irvine library; that first statement is flat wrong.

There are a lot of good things in here. The fundamentals in the first two chapters (numbering, hex/bin/octal conversion, 2s-complement, register structures, etc) still hold true. There are some organizational peculiarities but I attribute that to the author using a 15-20 year old book as his foundation.

I can't say this is a terrible book. In its time, it was probably a decent book. It's obvious that Irvine knows his stuff; I think most of my complaints can be attributed to the fact that this book should have been completely rewritten years (maybe a decade) ago. If you're a student and this is required text...you'll probably be okay. However, I'd hesitate recommending it as a self-study guide due to its price, datedness, & at times poor organization. Read more &rsaquo;

This book contains a large amount of 16 bit (and 32 bit code that is glorified 16 bit) x86 code. We are now on 64 bits. 64 bit architecture has also changed. This book is representative of the approximately the 2000-2002 era of Windows ME or so, despite the fact it keeps getting "updated". I do not know what these updates consist of, but it still fails to address modern operating systems and architecture. Many students, unfamiliar with DOS, will be quickly lost by the extensive use of the DOS console.

You will not learn x86 Assembly Language from this book. The pace is glacial and you will barely be able to output to the console by the time you finish. Assembly at this introductory level is not that hard, the author has just loaded down his book with useless minutiae, presumably to hide the fact that the book is increasingly outdated, its code is irrelevant, it fails to address the windows api, and it doesn't deal w/any modern MASM implementation. As a bonus, the writing is terrible and the programming exercises frequently require knowledge of material not covered at the time of the exercise (most of the chapter 5 exercises, for instance, essentially require bit-shifting, conveniently located in chapter 7 - 2 chapters afterwards), or, still better, are often not covered at all.
But, perhaps the worst part of the book is that the author does not use a standard library (MASM32 would be ideal - or the libraries included w/the latest Visual Studio Express - available for free) instead the author writes his own proprietary library for the outdated last pre-VS hurrah of MASM. So, when you're finished with the book, not only will you have learned little, but you will not even have learned many standard calls or even a modern MASM implementation. Computers progress quickly. This author hasn't. I can't possibly emphasize how outdated this text is. Reading this is basically like registering for an organic chemistry class and then being taught about the wonders of phlogiston by an alchemist.

Try instead: Professional Assembly Language (Programmer to Programmer), Detmer, Dandamundi, or read any of the million tutorials on the MASM32 website. If you can code already, the Visual C++ Optimization with Assembly Code is quite good and written by someone who actually programmed at some point after the Reagan administration.

I teach Assembly Language at a community college, and have used the third edition with much success. Matter of fact, I was so impressed by it that when Dr. Irvine asked if any of the current teachers who'd registered at his site wanted to review the fourth edition, which was still being written, I volunteered. The fourth edition is an organizational improvement on an already well-written book.

Unlike many Computer Science books purported to be for beginners, this one really is written for students near the beginning of their studies. It's aimed at those who have taken maybe one previous programming class. Dr. Irvine has a knack for reducing a very complex topic to its elements and explaining those elements so they're easily understood.

The book description states that it is “Designed for students and professionals interested in learning the basics of operating systems and architecture in the context of a microprocessor.” Although the book covers some of the critical topics necessary for operating systems programming on the Intel Architecture, e.g. I/O, segmentation, interrupts, it fails to do so adequately, and it omits discussion of protection, paging, and privileged mode instructions, to name three among many. I grant that you could learn enough to understand the basics of real-mode DOS, but not any of the Windows flavors, OS/2, nor the varieties of Minix/Linux.

I have recently completed a university course in Assembly Language programming. The instructor opted to use this book as the text for the course. First, the book is riddled with typos that are serious errors which can confuse a student into a state of mindless frustration. I would like to write a lengthy review, but for the sake of brevity, I can tell all in the few words that follow. In short, I shelved the book during the duration of the course and read "Peter Norton's Assembly Language Guide." I received a 4.0 (A) in the course which others who continued using the Irvine book struggled to make a passing mark. If all the errors are corrected in the text, this might be a worthwhile reference, but as it stands, it's more damaging than helpful. No offense intended for the author, just a bit of constructive criticism.

This text addresses the needs of students who can write programs in a high-level language and wish to learn assembly language for the Intel family of microprocessors. The forth edition contains an abundance of new material. The author introduces 32-bit protect-mode programming using the Microsoft Windows platform and he delays coverage of 16-bit real-mode programming using the MS-DOS platform until chapter 13. Only MS-DOS programming was covered in the previous edition. Each section contains review questions and each chapter ends with a summary and several programming exercises. The author clearly describes each concept and uses excellent diagrams and code examples throughout the text. Several new topics have been added to this edition such as graphics programming in both Windows and DOS applications.

http://eduln.org/2948.pdf
http://eduln.org/2779.pdf
http://eduln.org/1169.pdf
http://eduln.org/1526.pdf
http://eduln.org/2295.pdf