汇编语言

基础教程

HUIBIAN YUYAN
JICHU JIAOCHENG

主　编／谈文蓉
副主编／姜　玥

软硬结合　层次递进
任务驱动　图文并茂

西南交通大学出版社

普通高等学校“十三五”通用性基础规划教材
西南民族大学教材建设基金资助出版

汇编语言基础教程

主　编　谈文蓉
副主编　姜　玥

西南交通大学出版社
·成　都·

内容简介

汇编语言可以利用 CPU 指令直接控制计算机硬件，是初学者理解计算机硬件工作原理和高级程序语言功能、运行机理的最佳工具。不同类型的计算机架构与 CPU，对应着不同的汇编语言。本书详细介绍了三大主流处理器架构 MIPS、x86 和 ARM 在体系结构、指令系统、应用领域等方面的不同，帮助读者深入了解汇编语言的底层属性。本书以 x86 架构和 8086 汇编语言作为实例，来驱动汇编基础知识的教学：借助 DEBUG 调试工具帮助读者边学习边实践来理解寄存器、存储器、指令格式、寻址方式和机器代码等概念，借助 MASM 开发工具帮助读者掌握汇编语言软件的开发环境及开发过程，运用 CPU 指令和对应的开发工具进行汇编语言程序设计。

本书语言易懂、结构清晰，循序渐进地展开讲解、安排实验。本书可作为高等院校计算机、电子信息、自动化、通信类专业的教材（含实验）或参考书，是广大读者学习计算机组成原理、接口技术、单片机应用、嵌入式系统等专业课程的基础。

图书在版编目（CIP）数据

汇编语言基础教程 / 谈文蓉主编. —成都：西南交通大学出版社，2016.7
普通高等学校“十三五”通用性基础规划教材
ISBN 978-7-5643-4763-5

Ⅰ. ①汇… Ⅱ. ①谈… Ⅲ. ①汇编语言 - 程序设计 - 高等学校 - 教材 Ⅳ. ①TP313

中国版本图书馆 CIP 数据核字（2016）第 154651 号

普通高等学校“十三五”通用性基础规划教材

汇编语言基础教程

主编　谈文蓉

责任编辑	黄庆斌
特邀编辑	秦明峰
封面设计	墨创文化
出版发行	西南交通大学出版社 （四川省成都市二环路北一段 111 号 西南交通大学创新大厦 21 楼）
发行部电话	028-87600564　028-87600533
邮政编码	610031
网址	http://www.xnjdcbs.com
印刷	成都中铁二局永经堂印务有限责任公司
成品尺寸	185 mm× 260 mm
印张	11
字数	233 千
版次	2016 年 7 月第 1 版
印次	2016 年 7 月第 1 次
书号	ISBN 978-7-5643-4763-5
定价	28.00 元

课件咨询电话：028-87600533
图书如有印装质量问题　本社负责退换

前　言

近年来，人们经常认为汇编语言的应用范围很小，而忽视它的重要性。事实上，汇编语言是学习计算机技术的重要基础。通过学习和使用汇编语言，能够感知、体会、理解机器的逻辑功能，向上可以理解各种软件系统，向下能够感知硬件；充分获取计算机底层的编程经验，深刻理解计算机程序的运行机制。

作为低级语言，汇编语言与处理器架构、指令系统密切相关。写程序虽然不是汇编语言的强项，但却是理解计算机硬件工作原理与计算机程序执行的重要途径之一，学习汇编语言的两个目的：体验底层编程和深刻理解机器运行程序的机理。

目前，X86 架构的处理器占据了超过 90%的个人计算机市场，以 ARM 架构为代表的 RISC 产品则占据了超过 90%的移动计算机市场。在 MIPS、X86 和 ARM 三大主流处理器架构中选择基于 X86 架构的 8086 CPU 进行汇编语言的入门教学，是出于这样两个方面的考虑：

（1）8086 CPU 的汇编语言相对简单、学习门槛低，方便教与学。

（2）任何一台个人计算机均可以模拟 8086 的运行环境，方便随时随地实践。

学习汇编语言就是要紧密接触底层，要保持它的“原汁原味”。本书以 Intel 8086 指令系统和 DEBUG 为主体，在个人计算机的 MS-DOS 和 Windows 操作系统平台中，循序渐进地展开实验。DEBUG 提供了用户与计算机内部联系的窗口，利用调试器 DEBUG，可以深入到机器内部，观察到 CPU 和存储单元的情况；可以用单步执行命令跟踪执行，每执行一步都能使用户看到各寄存器的内容变化，以便分析和调试程序。

汇编语言教学必须强调与实际机器结合。为了更好地引导、帮助读者理解硬件系统，作者精心创作本书。本书的特点是：简明实用，学生按该书的介绍就能掌握基本理论知识直接上机。对于学习一门技术而言，模仿是快速掌握技能的一个捷径。本书对示例程序的分析以 DEBUG 下的操作和运行结果为依据，读者有样板可学，有结果可见。

本书特别强调动手训练，在每章都采用边讲解理论边练习的方式。读者通过学习和实践，能为下一步学习微机原理和接口技术课程、单片机原理和应用、嵌入式课程和计算机组成原理等课程打下良好基础，培养了计算机系统能力。

本书的内容涵盖理论教学和实验教学，与现有众多教材相比，有以下特色：

（1）从不同处理器架构在体系结构、指令系统、应用领域上的不同入手，帮助读者建立对汇编语言底层属性的感性认识。

（2）将枯燥的理论教学与繁杂的实践教学紧密结合，每章都采用边学习边实践边理解的方式，大大提高了学习效率和效果。

本书是作者长期开展汇编语言课程教学改革的成果，得到了四川省教学质量工程项目，西南民族大学教材建设基金项目及西南民族大学教育教学改革项目(2015QN14)的资助。教材在编写过程中，得到相关老师和同学的帮助，也参考了其他同行的教材，在此深表感谢。

本书由谈文蓉教授作为主编，姜玥副教授作为副主编，谢盈和郑海春(西华大学)老师参与编写。书中的全部程序段都经过调试，如有错误和不当之处，欢迎读者批评指正。

编　者

2016 年 3 月

目　录

第 1 章　汇编语言基础知识

导读

本章介绍了学习汇编语言的目的和计算机中数据的表示方法，主要包括计算机程序设计语言、汇编语言的特点，带符号数的补码表示和 ASCII 码。

主要知识点

- 计算机程序设计语言
- 汇编语言的特点
- 带符号数的补码表示
- ASCII 码

1.1　为什么学习汇编语言

学习汇编语言，是学习计算机科学与技术的最佳起点。正如学习美国文化要先懂英语一样，如果不了解计算机的语言，又何谈懂计算机呢？为了解决这个问题，这要从 CPU 的指令特点和计算机程序设计语言的发展史两个方面来考虑。

1.1.1　CPU 指令

在计算机中，指令和数据都用二进制来表示，也就是说它只认识 0 和 1 这样的数字。最早期的计算机程序通过在纸带上打洞的人工操作的方式来模拟 0 和 1，根据不同的组合完成一些操作。后来将完全通过 0 和 1 编写程序的语言，称之为机器语言。为了让计算机知道这些组合的意思，于是就出现了 CPU 指令。

CPU 指令对应了 0 和 1 的一些组合。每款 CPU 在设计时就规定了一系列与其硬件电路相配合的指令系统。通过 CPU 指令集的文档，就可以编写 CPU 认识的机器代码了。所以对于不同 CPU 来说可能会有不同的机器码。比如下面定义了一套 CPU 硬件电路可以完成的 CPU 指令，如表 1.1 所示。随着计算机的发展，CPU 支持的指令也越来越多，功能也越来越强。

表 1.1　自定义的 CPU 指令

指令	格式	说明
0001	[address][register]	读取存储器的值到寄存器
0010	[register][address]	寄存器的值写入到存储器
0011	[register1][register2]	加法操作

1.1.2　计算机程序设计语言

计算机编程语言的发展经历了机器语言、汇编语言和高级语言。其中，前两种语言与机器密切相关，称为低级语言。作为计算机“母语”的低级语言，会伴随计算机本身永存于世。

1. 机器语言

计算机能够直接识别的是二进制数 0 和 1 组成的代码。机器语言是计算机唯一能够识别的语言，只有机器语言描述的程序，计算机才能直接执行。用其他语言编写的程序必须转换成机器语言程序，即目标程序。目标程序就是为源程序经编译可直接被计算机运行的机器码集合，在计算机文件上以“.obj”作扩展名。

例 1.1　实现两个 10 进制数据 100 和 256 相加的功能，在 8086CPU 上，机器语言编写的十六进制代码：

```
B8 64 00
05 00 01
```

区区两行代码，几乎没有人能够直接读懂。机器语言的缺点是难以记忆、表达和阅读。只有在计算机诞生的初期，才用机器语言编程。

2. 汇编语言

为了克服机器语言的缺点，人们将机器指令符号化，用描述指令功能的符号来表示机器指令，这些符号称为指令助记符。

用助记符表示的指令就是汇编格式指令。汇编格式指令及使用它们编写程序的规则形成汇编语言。汇编语言编写的程序就是汇编语言源程序，扩展名为“.asm”。汇编程序将汇编语言程序“汇编”成机器代码目标模块的程序，扩展名为“.obj”。

汇编语言是最接近于机器语言的编程语言。机器语言是计算机操作的本质，汇编语言是最接近本质的语言。

例 1.2　实现两个 10 进制数据 100 和 256 相加的功能，用 8086 CPU 的汇编语言编写的代码：

```
MOV AX,64       ;AX←64H
ADD AX,100      ;AX←AX+100H
```

编写汇编语言源程序，必须了解计算机内部可供使用的资源，熟悉汇编指令的细节，安排运算的每一个步骤，这使得编写汇编语言源程序的过程细腻、繁杂；但麻烦之后带给用户的是能够直接操控计算机硬件的愉悦感。通过汇编语言能够直接感知计算机内部各部件之间的相互作用。

3. 高级语言

汇编语言较机器语言直观，但和具体的计算硬件关系密切。高级语言比较接近于人类自然语言的语法习惯及数学表达形式，一般计算机用户不需要懂得计算机的结构和工作原理。高级语言屏蔽了计算机内部实现每一个程序的具体细节，这使得高级语言的语句看上去简单清晰。由于省略了很多细节，用高级语言编程不需要有太多的计算机专业知识。高级语言有很多，例如 C/C++、JAVA、VB，都是比较常用的。

计算机执行的语言，或者称之为命令序列或数据，都是为 1 和 0 的二进制语言，物理上则表现为电信号的高低电平。虽然我们现在有 C/C++、Java、Python 等一系列强大的高级语言，但是其真正落实到计算机的执行时还是需要编译或解释成计算机懂的机器语言。

例 1.3　实现两个 10 进制数据 100 和 256 相加的功能，用 C 语言编写的代码：

```
X=100+256
```

从以上的程序设计语言可以看出，只有通过汇编语言指令才能正确全面地了解计算机的基本功能和行为方式；任何其他编程语言都必须编译成机器语言（本质上也可以说是汇编语言）代码才能被计算机接受和执行，所以，汇编语言在计算机中居于顶（软件之）天立（硬件之）地的重要地位。借助汇编语言从 CPU 的层面思考问题，有效地提高计算机科研及应用开发的思维深度；任何高级语言都必须翻译成机器（或汇编）语言才能执行，所以任何高级语言的功能和实现机理，最终都将以机器（或汇编）代码的形式 —— 简明无二义性地表述出来。也就是说，通过反汇编代码可以透析和研究任何高级语言的功能和实现机理。汇编语言则似乎是透析高级语言功能机理的有效工具。

1.1.3　汇编语言的特点

一方面，汇编语言指令是用一些具有相应含义的助记符来表达的，所以，它要比机器语言容易掌握和运用。另一方面，它要直接使用 CPU 的资源，相对高级程序设计语言来说，它又显得难掌握。

汇编语言程序归纳起来大概有以下几个主要特性。

1. 与机器相关性

汇编语言指令是机器指令的一种符号表示，而不同类型的 CPU 有不同的机器指令

系统，也就有不同的汇编语言，所以，汇编语言程序与机器有着密切的关系。

由于汇编语言程序与机器的相关性，所以，除了同系列、同型号 CPU 之间的汇编语言程序有一定程度的可移植性之外，其他不同类型 CPU 之间的汇编语言程序是无法移植的，也就是说，汇编语言程序的通用性和可移植性要比高级语言程序低。

2. 执行的高效率

汇编语言保持了机器语言的优点，具有直接和简捷的特点，可有效地访问、控制计算机的各种硬件设备，如磁盘、存储器、CPU、I/O 端口等，且占用内存少，执行速度快，是高效的程序设计语言。

3. 编写程序和调试的复杂性

由于是直接控制硬件，且简单的任务也需要很多汇编语言语句，因此在进行程序设计时必须面面俱到，需要考虑到一切可能的问题，以合理调配和使用各种软、硬件资源。这样，就不可避免地加重了程序员的负担。与此相同，在程序调试时，一旦程序的运行出了问题，就很难发现。

1.1.4 学习汇编语言的主要目的

学习汇编语言，其原因至少有以下几点：

（1）一些特定的场合需要使用汇编语言。

① 对软件的执行时间或存储容量要求较高的场合，如系统程序的关键核心等。

② 软件与硬件关系紧密，软件要直接控制硬件的场合，如设备驱动程序。

③ 没有合适高级语言的场合。

（2）有助于深入地理解计算机硬件，掌握计算机硬件、操作系统、应用程序之间的交互工作。

汇编语言操作直接面向硬件，指令操作更直接，通过一条一条直接控制计算机的指令，清晰地看到计算机的工作，理解计算机的内部工作方式，从而对计算机硬件和应用程序之间的联系和交互形成一个清晰的认识，形成一个软、硬兼备的编程知识体系。例如，CPU、内存和硬盘等硬件设备如何协调地工作在一起，数据从哪里转移到哪里，在哪里运算和存储等。

（3）帮助加深对高级语言的理解。

虽然现在有许多的高级语言可以用来编程，但是要真正理解代码执行的实际过程，从本质上理解机器的行为，需要理解汇编指令的执行。例如，C 语言中的指针概念，就是内存的地址。汇编语言对于内存的操作都是基于内存地址的。指针的学习和应用就是在指针这个抽象的概念和实际的内存单元之间建立思维映射，而这恰恰是在汇编语言中频繁操作的。

（4）是学习后续专业课程的基础。

汇编语言课程是计算机组成原理、接口技术和嵌入式系统等的先修课程。

1.2 数据表示

汇编语言涉及机器硬件层面，要从机器的角度以二进制和十六进制的思维考虑问题。在计算机中，常采用数的符号和数值一起编码的方法来表示数据。常用的有原码、反码和补码等，这几种表示法都将数据的符号数码化。为了区分一般书写时表示的数和机器中 0 和 1 编码表示的数，称前者为真值，后者为机器数。

1.2.1 带符号数的补码表示

1. 原　码

最高有效位表示符号（0 表示正数，1 表示负数），其他位表示数值。在计算机中用原码作加减运算是不方便的。

例 1.4　$[106]_{原码}$ = **0** 1101010B

$[-106]_{原码}$ = **1** 1101010B

2. 反　码

最高有效位表示符号（0 表示正数，1 表示负数），其他位表示数值。正数的反码与正数的原码相同；负数的反码是正数的原码（包括符号位）按位取反。

例 1.5　$[106]_{反码} = [106]_{原码}$ = 0 1101010B

$[-106]_{反码} = [106_{原码}]_{求反}$=1 0010101B

3. 补　码

正数的补码表示与正数的原码相同，即最高符号位用 0 表示正，其余位为数值位。负数的补码最高符号位用 1 表示负数。负数的补码表示是正数的补码各位求反（包括符号位），末位加 1 构成。将负数的补码转换成真值，则将补码末位减 1，各位求反，再加上负号。

例 1.6　X=106=0110 1010B，$[X]_{补}$=0110 1010B

X= - 106，$[X]_{补}=[106_{补码}]_{求反}+1=[0110\ 1010B]_{求反}+1=1001\ 0101B+1=1001\ 0110B$

例 1.7　求补码 1001 0110B 的真值。

分析：补码 1001 0110B 的最高位为 1，说明是负数的补码，将例 1.6 的过程反过来做，即减 1 后，各位求反，最后加上负号。

$[1001\ 0110B - 1]_{求反}=[1001\ 0101B]_{求反}$=0110 1010B=106

则[1001 0110B]$_{真值}$= - 106

计算机中，带符号数默认采用补码表示，因为利用补码进行减法运算比原码方便。使用补码可以带来以下好处：加法和减法可以用同一个运算器实现；无符号数和带符号数的运算可以用同一个运算器实现，从而大大简化计算机运算器电路，简化指令系统。n 位二进制数可以表示的无符号数范围：0 ~ 2^n - 1；n 位二进制补码表示的带符号数范围：- 2^{n-1} ~ 2^{n-1} - 1。

1.2.2 ASCII 码

字母和各种字符必须按特定的规则用二进制编码才能在计算机中表示，最常用的编码是 ASCII（America Standard Code for Information Interchange）码，即美国信息交换标准码。标准 ASCII 码见表 1.2。

表 1.2 标准 ASCII 码及字符

低 4 位	高 4 位							
	0 -	1 -	2—	3—	4—	5—	6—	7 -
- 0	NUL	DLE	SP	0	@	P	′	p
- 1	SOH	DC1	!	1	A	Q	a	q
- 2	STX	DC2	“	2	B	R	b	r
- 3	ETX	DC3	#	3	C	S	c	s
- 4	EOT	DC4	$	4	D	T	d	t
- 5	ENQ	NAK	%	5	E	U	e	u
- 6	ACK	SYN	&	6	F	V	f	v
- 7	BEL	ETB	’	7	G	W	g	w
- 8	BS	CAN	(	8	H	X	h	x
- 9	HT	EM	)	9	I	Y	i	y
—A	LF	SUB	*	:	J	Z	j	z
—B	VT	ESC	+	;	K	[	k	{
- C	FF	FS	,	<	L	\	l	|
—D	CR	GS	-	=	M	]	m	}
- E	SO	RS	.	>	N	^	n	~
—F	SI	US	/	?	O	-	o	DEL

说明：编码是十六进制，其中 SP 表示空格，BEL 表示振铃，LF 表示换行，CR 表示回车。

例 1.8 ① 数字值 0 ~ 9 和数字字符'0' ~ '9'的转换关系；② 十六进制数 0AH ~ 0FH

和字符'A'~'F'的转换关系。

分析：从表 1.2 可见，数字字符和数字值之间的转换关系通过观察数字字符对应的 ASCII 码与其数字值之间的关系找到。

（1）数字字符'0'~'9'对应的 ASCII 码：30H~39H，数字字符的 ASCII 码和数字值之间相差了 30H，若数字字符向数字值转化（'0'~'9'→0~9），则数字字符的 ASCII 码减去 30H，就得到对应的数字值，即$[数字字符]_{ASCII}$ - 30H=数字值；

若数字值向数字字符转化（0~9→'0'~'9'），则数字值加上 30H，就得到对应的数字字符，数字值+30H=$[数字字符]_{ASCII}$。

（2）由于字符'A'~'F'对应的 ASCII 码：41H~46H，与十六进制数 0AH~0FH 相差了 37H，若数字字符向数字值转化（'A'~'F'→0AH~0FH），则数字字符的 ASCII 码减去 37H，就得到对应的数字值，即$['A'\sim'F']_{ASCII码}$ - 37H=0AH~0FH；

若数字值向数字字符转化（0AH~0FH→'A'~'F'），则数字值的 ASCII 码加上 37H，就得到对应的数字字符，即(0AH~0FH) +37H=$['A'\sim'F']_{ASCII码}$。

例 1.9　大小写字母之间的转换关系。

分析：从表 1.2 可见，大小写字母之间的转换关系通过观察大小写字母对应的 ASCII 码值找到。大写字母'A'~'Z'的 ASCII 码：41H~5AH；小写字母'a'~'z'的 ASCII 码：61H~7AH。大小写字母之间的 ASCII 码相差了 20H。所以大小写字母之间的转换即为在其对应的 ASCII 码基础上加减 20H。

若大写字母向小写字母转化（'A'~'Z'→'a'~'z'），则大写字母的 ASCII 码加上 20H，就得到小写字母的 ASCII 码，即$[大写字母]_{ASCII码}$+20H= $[小写字母]_{ASCII}$；

若小写字母向大写字母转化（'a'~'z'→'A'~'Z'），则小写字母的 ASCII 码减去 20H，就得到大写字母的 ASCII 码，即$[小写字母]_{ASCII码}$ - 20H= $[大写字母]_{ASCII}$。

小　结

计算机程序设计语言包括机器语言、汇编语言和高级语言。

汇编语言是机器语言的符号表示，与机器语言无本质区别。学习汇编语言，可以加深对计算机系统以及程序工作原理的理解。

计算机系统采用二进制表示数据。为了描述方便，经常采用十六进制形式。计算机中，带符号数默认采用补码表示。

习　题

1. 简述计算机程序设计语言的分类。
2. 汇编语言的特点。
3. 将下列十进制数分别用 8 位二进制数的原码、反码和补码表示：

（1）-120；（2）120；（3）-38。

4．计算机中有一个“01100000”编码，如果把它认为是无符号数，它是十进制什么数？如果把它认为是带符号数，它是十进制什么数？如果它是某个ASCII码，则代表哪个字符？

第 2 章　存储程序的计算机和指令系统结构

导读

本章介绍了存储程序的计算机结构和指令系统的结构，主要包括冯· 诺依曼结构和哈佛结构，CISC 和 RISC。

主要知识点

- 存储程序的计算机结构
- 指令系统结构

2.1　存储程序的计算机结构

存储程序和程序控制原理的要点是，程序输入到计算机中，存储在内存储器中（存储原理），在运行时，控制器按地址顺序取出存放在内存储器中的指令（按地址顺序访问指令），然后分析指令，执行指令的功能，遇到转移指令时，则转移到转移地址，再按地址顺序访问指令（程序控制）。

计算机是一种能够按照事先存储的程序，自动、高速地进行大量数值计算和各种信息处理的现代化智能电子设备。存储程序的计算机是一种计算机系统设计模型。冯· 诺依曼结构和哈佛结构分别都是一种存储器结构。

2.1.1　冯· 诺依曼结构

计算机是自动化信息处理装置，它采用“存储程序”工作原理，该原理是由美籍匈牙利数学家冯· 诺依曼 1946 年提出的。冯· 诺依曼结构是一种将程序指令存储器和数据存储器合并在一起的存储器结构。程序指令存储地址和数据存储地址指向同一个存储器的不同物理位置，因此程序指令和数据的宽度相同。使用这种概念和原理设计的电子计算机系统统称为“冯· 诺依曼结构”计算机。

冯· 诺依曼结构的设计思想简要地概括为以下三点：

（1）计算机包括运算器、存储器、控制器、输入和输出设备五大基本部件，如图 2.1 所示，其中虚线表示数据线，实线表示控制线。

（2）计算机内部采用二进制来表示指令和数据。