21世纪高等学校规划教材 | 计算机科学与技术

汇编语言程序设计教程

陆 遥 编著

清华大学出版社

汇编语言程序设计教程

陆 遥 编著

清华大学出版社
北京

内 容 简 介

本书讲授 Intel 8086 微处理器的指令系统，并以 Microsoft 的 MASM 5.0 版本宏汇编语言为基础，讲授汇编语言程序设计的基本方法和常用技术。

全书共分 5 章。第 1 章讲授学习汇编语言程序设计所需具备的基础知识，包括指令的概念、数据的表示、数据的存储和处理等；第 2 章讲授 8086 宏汇编语言的源程序组成，包括汇编语言的语言成分，常量、变量、标号等的定义，源程序的结构及定义等；第 3 章讲授 8086 的指令系统，包括寻址方式和各类操作指令等；第 4 章讲授 8086 汇编语言程序设计的基本方法，包括顺序程序、分支程序、循环程序、子程序、宏指令等；第 5 章介绍 8086 的中断技术，包括中断的相关概念，中断服务程序设计方法等。

本书可作为高等院校计算机及相关专业的汇编语言课程教材，也可作为从事计算机工作的专业人员的参考书。

图书在版编目（CIP）数据

汇编语言程序设计教程 / 陆遥编著. —北京：清华大学出版社，2018
（21 世纪高等学校规划教材 • 计算机科学与技术）
ISBN 978-7-302-49860-5

Ⅰ. ①汇… Ⅱ. ①陆… 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（2018）第 051386 号

责任编辑：郑寅堃
封面设计：傅瑞学
责任校对：李建庄
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn，010-62795954
印 装 者：三河市金元印装有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　张**：11.25　　**字　　数**：275 千字
版　　次：2018 年 9 月第 1 版　　**印　　次**：2018 年 9 月第 1 次印刷
印　　数：1～1000
定　　价：35.00 元

产品编号：078505-01

出版说明

随着我国改革开放的进一步深化，高等教育也得到了快速发展，各地高校紧密结合地方经济建设发展需要，科学运用市场调节机制，加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度，通过教育改革合理调整和配置了教育资源，优化了传统学科专业，积极为地方经济建设输送人才，为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是，高等教育质量还需要进一步提高以适应经济社会发展的需要，不少高校的专业设置和结构不尽合理，教师队伍整体素质亟待提高，人才培养模式、教学内容和方法需要进一步转变，学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007 年 1 月，教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》，计划实施“高等学校本科教学质量与教学改革工程（简称‘质量工程’）”，通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容，进一步深化高等学校教学改革，提高人才培养的能力和水平，更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中，各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势，对其特色专业及特色课程（群）加以规划、整理和总结，更新教学内容、改革课程体系，建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上，经教育部相关教学指导委员会专家的指导和建议，清华大学出版社在多个领域精选各高校的特色课程，分别规划出版系列教材，以配合“质量工程”的实施，满足各高校教学质量和教学改革的需要。

为了深入贯彻落实教育部《关于加强高等学校本科教学工作，提高教学质量的若干意见》精神，紧密配合教育部已经启动的“高等学校教学质量与教学改革工程精品课程建设工作”，在有关专家、教授的倡议和有关部门的大力支持下，我们组织并成立了“清华大学出版社教材编审委员会”（以下简称“编委会”），旨在配合教育部制定精品课程教材的出版规划，讨论并实施精品课程教材的编写与出版工作。“编委会”成员皆来自全国各类高等学校教学与科研第一线的骨干教师，其中许多教师为各校相关院、系主管教学的院长或系主任。

按照教育部的要求，“编委会”一致认为，精品课程的建设工作从开始就要坚持高标准、严要求，处于一个比较高的起点上；精品课程教材应该能够反映各高校教学改革与课程建设的需要，要有特色风格、有创新性（新体系、新内容、新手段、新思路，教材的内容体系有较高的科学创新、技术创新和理念创新的含量）、先进性（对原有的学科体系有实质性的改革和发展，顺应并符合 21 世纪教学发展的规律，代表并引领课程发展的趋势和方向）、示范性（教材所体现的课程体系具有较广泛的辐射性和示范性）和一定的前瞻性。教材由个人申报或各校推荐（通过所在高校的“编委会”成员推荐），经“编委会”认真评审，最后由清华大学出版社审定出版。

目前，针对计算机类和电子信息类相关专业成立了两个“编委会”，即“清华大学出版社计算机教材编审委员会”和“清华大学出版社电子信息教材编审委员会”。推出的特色精品教材包括：

（1）21 世纪高等学校规划教材·计算机应用——高等学校各类专业，特别是非计算机专业的计算机应用类教材。

（2）21 世纪高等学校规划教材·计算机科学与技术——高等学校计算机相关专业的教材。

（3）21 世纪高等学校规划教材·电子信息——高等学校电子信息相关专业的教材。

（4）21 世纪高等学校规划教材·软件工程——高等学校软件工程相关专业的教材。

（5）21 世纪高等学校规划教材·信息管理与信息系统。

（6）21 世纪高等学校规划教材·财经管理与应用。

（7）21 世纪高等学校规划教材·电子商务。

（8）21 世纪高等学校规划教材·物联网。

清华大学出版社经过三十多年的努力，在教材尤其是计算机和电子信息类专业教材出版方面树立了权威品牌，为我国的高等教育事业做出了重要贡献。清华版教材形成了技术准确、内容严谨的独特风格，这种风格将延续并反映在特色精品教材的建设中。

清华大学出版社教材编审委员会
联系人：魏江江
E-mail:weijj@tup.tsinghua.edu.cn

前言

汇编语言是一种低级语言，其程序设计需要涉及计算机的数据表示、寄存器的使用方式、存储器的访问方式、输入/输出的实现方式等与计算机硬件相关的知识和技术。汇编语言也是一种典型的面向过程的程序设计语言，编程者必须全面细致地把握和控制问题处理的全过程，才能设计出好的程序。

汇编语言程序设计是计算机专业的一门重要的专业课程。就课程地位而言，它处于硬件课程和软件课程的结合部，与硬件和软件都有着密切的关系。汇编语言是学生了解计算机硬件及其工作原理的入口，是计算机组成原理、微机接口技术、单片机应用技术、嵌入式系统等涉及硬件原理与应用技术课程的基础；同时，汇编语言程序设计能很好地培养和锻炼学生的程序设计能力，从而夯实学生的软件设计基础。

笔者在多年的汇编语言程序设计课程教学中，接触过不少相关的教材，但始终难觅真正适合当前教学要求的好教材。一些知名教材，也存在工具书化和手册化严重的情况，其他同类教材也基本趋同。

鉴于此，笔者尝试以自己多年的教学积累和在汇编语言应用实践方面（如图像处理、病毒查杀、硬件控制等）的实际经验为基础，以同类型优秀教材和文献资料为参考，编写一本满足当前汇编语言教学实际需要的教材。本书的主要特点有：

（1）应用性突出。计算机语言是用来编写程序解决问题的。本书用丰富的实例和详细的解释，突出汇编语言的编程应用技术，其中有很多实例提供了汇编语言编程应用中颇具实用价值的解决方案。

（2）内容取舍有度。本书的编写充分结合当前汇编语言教学的实际需要，不求全，不追求工具书化和手册化，一切从实用出发，从满足教学需要出发，对内容进行了精选和提炼，使全书的内容更加精练，重点更加突出，应用于教学更加顺畅。

（3）讲解详细到位，可读性好。本书杜绝简单的内容罗列，对所讲的内容必详细阐述，必要时辅以实例。本书力争用通俗易懂的文字来描述各种专业性的概念和问题，以便读者更好地理解书中的内容。

（4）习题设计突出应用性。本书在习题设计上摒弃大量的概念、语法类习题，而是以提高编程应用能力为目的，由浅入深，由易到难，设计了各种应用型习题。

此外，本书在内容组织上，将汇编语言的源程序组成置于指令系统之前。这是有别于其他教材的创新点。这样的安排可以使读者尽早建立起汇编语言源程序的整体结构概念，有利于尽早开展应用编程和上机实践。这也是本书突出应用性的体现。

程序设计课程十分强调上机编程实践。本书在附录中，详细介绍了汇编语言的上机环境和主要工具软件的使用方法，以期有效地指导读者上机。

笔者期待自己在本书中所做的尝试和努力能够得到读者朋友们的认可，也恳请读者朋

友对本书提出宝贵的意见和建议，共同为这门课程教学质量的提高而努力。笔者的电子邮箱地址：1305413741@qq.com。

为方便本课程的教学，本书为授课教师准备了课程电子教案和习题参考解答，如有需要，请与清华大学出版社编辑郑寅堃（ZhengYK@tup.tsinghua.edu.cn）联系。

陆 遥

2018 年 6 月于桂林

目 录

第1章 基础知识

1.1 汇编语言的特点

1.1.1 汇编语言与机器语言的关系

计算机的程序设计语言（简称计算机语言）是人们用来给计算机描述操作任务的工具。

由于计算机是一种数字逻辑设备，它只能识别用二进制代码表示的信息，所以，最初的计算机语言是直接用二进制代码来表述的，这就是机器语言。机器语言的基本要素是机器指令（简称指令），每条指令用于给计算机下达一个简单操作任务，一个复杂的解题任务需要按一定的顺序执行多条指令才能完成。这种按一定顺序排列起来的指令序列就是程序。

机器语言的优点是程序执行速度快、占用存储空间小，缺点是语言难以掌握，程序调试和排错困难，需要对计算机的硬件系统有较多的了解。

为了便于掌握和使用，人们将机器语言符号化，产生了汇编语言。汇编语言使用一些容易掌握和使用的符号来表示每条指令，使编程和调试更加方便。

例如，在 8086 系统中，以下机器指令代码

```
0000000111011000
```

所描述的操作为：将 0 号 16 位寄存器中的数据与 3 号 16 位寄存器中的数据相加，其和存入 0 号 16 位寄存器。这串二进制代码中，各部分所代表的含义如表 1.1 所示。

表 1.1 机器指令代码分析

00000001	11	011	000
描述操作性质，即将两个 16 位寄存器中的数据相加	表示两个数据均取自寄存器	描述 3 号寄存器	描述 0 号寄存器

显然，要熟练掌握和使用这种机器指令代码是很困难的。而汇编语言则把一条机器指令的操作性质和操作对象分别用符号表示。如在 8086 系统的汇编语言中，上述指令被符号化为：

```
ADD  AX , BX
```

其中，ADD 代表相加操作，AX 代表 0 号 16 位寄存器，BX 代表 3 号 16 位寄存器。汇编

语言指令中所使用的这些符号称为助记符，这种符号化的汇编语言指令显然更容易掌握和使用。

由于计算机不能直接理解汇编语言的符号系统，所以，需要一个转换工具来将用汇编语言编写的程序转换成机器语言程序，这个转换工具叫作汇编程序。

1.1.2 汇编语言与高级语言的主要差异

计算机的程序设计语言分为低级语言和高级语言两大类，其中，机器语言和汇编语言属于低级语言，其余均为高级语言。语言的“高级”与“低级”之分，并不是指语言之间的优劣，而是指语言的使用是否直接涉及计算机的硬件。高级语言在使用过程中，不用（或基本上不用）直接与计算机的硬件打交道，而使用低级语言则时刻需要直接操作计算机的硬件。

例如，用汇编语言编程时，必须准确指出数据存放的地方——某个寄存器、某个存储单元或某个 I/O 端口，必须直接控制相关的设备完成数据的输入/输出。而用高级语言编程时，则无须关心一个数据究竟是存放在寄存器中，还是存放在内存中，而当需要输入或输出数据时，只要写出一条输入或输出语句即可，不用直接去控制相关的输入/输出设备。

相对于汇编语言，高级语言更便于描述复杂的程序控制结构及处理功能，更接近人们的语言习惯，并且基本上不直接涉及计算机硬件概念，所以更容易掌握和使用。但用任何一种高级语言编写的程序，都必须转换成机器语言程序才能被计算机执行。完成这种转换任务的工具叫作编译程序，每种高级语言都要配备自己的编译程序。

直接用汇编语言编程虽然困难一些，但编出的程序时、空效率高（即运行速度快，占用存储空间少）；而用高级语言编写的程序由编译程序转换为机器代码后，并不是最优化的执行代码，其时、空效率要低得多。此外，在需要直接控制计算机硬件的应用场合，汇编语言比高级语言更灵活、方便，甚至是非用汇编语言不可的。

因此，汇编语言与高级语言各有其应用场合，学习和掌握汇编语言程序设计方法，是提高计算机应用能力的重要基础。

值得指出的是，低级语言是与计算机硬件系统的功能设计、组成结构密切相关的，因此，不同系列的计算机，其低级语言是不兼容的。尽管如此，其汇编语言的基本特点及程序设计的基本方法是相通的。

1.2 计算机中的数据表示

由于汇编语言要涉及数据存储和处理的细节，所以只有掌握了计算机中的数据表示，才能正确、有效地进行数据处理。

在计算机内部，任何类型的数据均以二进制数字序列编码表示。但是，直接用二进制表示数据位数多，较为烦琐，因此，无论是在书面上，还是在用汇编语言编程时，均允许使用其他进制的数据表示方法。几种常用数制的规定标志是：B 代表二进制，D 代表十进制，O 代表八进制，H 代表十六进制。使用时将这些标志字母附在所表示的数据后面即可，

如 10001001B，105D，36D4H 等。

1.2.1　字符数据表示

字符在计算机中的二进制编码称为字符代码。目前，计算机中普遍使用的字符代码是 ASCII 码（美国信息交换标准代码）。ASCII 字符集共包含 256 个字符，分成两个子集——基本字符集和扩展字符集，各包含 128 个字符。ASCII 码实际上就是这 256 个字符的编号，用 8 位二进制编码表示，范围是 00000000～11111111。

在 ASCII 字符集中，基本字符集包含了人们日常书面表达中所用的各种字符（也称可打印字符），如大小写英文字母、数字字符 0~9、基本运算符和标点符号、空格等，此外，还包含一些计算机内部用于控制的特殊字符，如回车、换行、删除、换码等。基本字符集是各类应用中主要使用的字符集，由 ASCII 字符集的前 128 个字符组成，编码范围是 00000000～01111111，如表 1.2 所示。

表 1.2　ASCII 基本字符集编码表

b_3 b_2 b_1 b_0 \ b_7 b_6 b_5 b_4	0000	0001	0010	0011	0100	0101	0110	0111
0 0 0 0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	SOH	DC_1	!	1	A	Q	a	q
0 0 1 0	STX	DC_2	“	2	B	R	b	r
0 0 1 1	ETX	DC_3	#	3	C	S	c	s
0 1 0 0	EOT	DC_4	$	4	D	T	d	t
0 1 0 1	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	ACK	SYN	&	6	F	V	f	v
0 1 1 1	BEL	ETB	‘	7	G	W	g	w
1 0 0 0	BS	CAN	(	8	H	X	h	x
1 0 0 1	HT	EM	)	9	I	Y	i	y
1 0 1 0	LF	SUB	*	:	J	Z	j	z
1 0 1 1	VT	ESC	+	;	K	[	k	{
1 1 0 0	FF	FS	,	<	L	\	l	|
1 1 0 1	CR	GS	–	=	M	]	m	}
1 1 1 0	SO	RS	.	>	N	^	n	~
1 1 1 1	SI	US	/	?	O	_	o	DEL

扩展字符集由 ASCII 字符集的后 128 个字符组成，包含一些图形符号和制表符等，编码范围是 10000000～11111111，一般较少使用。

熟悉表 1.2 中常用字符的 ASCII 码分布，对字符数据的处理是十分有利的。例如，从表中可知，数字字符 0～9 的 ASCII 码用十六进制表示为 30H～39H，与对应的数字之间相差 30H，这给我们在数字字符（简称“数字符”）与对应的数字之间的转换提供了依据；又如，大、小写英文字母的 ASCII 码范围分别是 41H～5AH 和 61H～7AH，即对应的大、小写字母的 ASCII 码之间相差 20H，这使我们知道如何进行大、小写字母之间的转换。

需要注意的是，由于通常所用的字符都是基本字符集中的字符，所以，很多书中所说的 ASCII 码只针对基本字符集，而这部分 ASCII 码的最高位均为 0，因此称 ASCII 码为 7

位编码（最高位上的 0 被忽略）。

一个字符的 ASCII 码在存储器中存放时，需要占用存储器的一个字节（8 位）。

1.2.2 数值数据表示

数值数据是计算机中用于各种算术运算的数据。数值数据在计算机中的表示方法有 BCD 码表示法、定点数表示法和浮点数表示法三种。

1. BCD 码

BCD（Binary Coded Decimal）码采用 4 位二进制编码表示 1 位十进制数，分有权码与无权码两类。计算机中实际使用的是一种有权 BCD 码——8421 码，其 4 位二进制编码的权值由高位到低位分别是 8、4、2、1，故名之。表 1.3 所列为十进制数 0～9 的 8421 码。

表 1.3 8421 码

十 进 制 数	8421 码	十 进 制 数	8421 码
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

8421 码可用于算术运算，其运算结果也用 8421 码表示，但是，只有在运算结果为正数时才是可接受的，且运算结果往往需要调整（或修正）后，才能形成正确的结果。下面以 8421 码加法运算为例进行说明。

【例 1.1】 用 8421 码计算 3+6。

解：

```
    0 0 1 1 …… 3 的 8421 码
+   0 1 1 0 …… 6 的 8421 码
-------------------------
    1 0 0 1 …… 9 的 8421 码
```

显然，本例的运算结果是正常的。

【例 1.2】 用 8421 码计算 5+7。

解：

```
    0 1 0 1 …… 5 的 8421 码
+   0 1 1 1 …… 7 的 8421 码
-------------------------
    1 1 0 0 …… 不在正常的 8421 码范围内
+   0 1 1 0 …… 对运算结果加 6 调整
-------------------------
  1 0 0 1 0 …… 十进制数 12 的 8421 码
```

本例对运算结果进行加 6 调整后，得到正确的运算结果 12。需要加 6 调整的原因是：就十进制而言，5 加 7 应该形成进位，但 8421 码是用 4 位二进制数表示的，需要满十六最高位上才会产生进位，因此需要加 6 促使进位产生。归纳起来，只要运算结果在 1010～1111 范围内，均需要加 6 调整。

【例 1.3】 用 8421 码计算 8+9。

解：

```
    1000 …… 8 的 8421 码
+   1001 …… 9 的 8421 码
  1 0001 …… 运算结果对应于十进制数 11，是错误的
+   0110 …… 对运算结果加 6 调整
  1 0111 …… 十进制数 17 的 8421 码，结果正确
```

本例的运算结果虽然产生了进位，但是在满十六的情况下进位的，而十进制只需满十即应进位，因此需要在低位上加 6 调整，才能符合十进制的进位规则。

如果用 8421 码进行多位十进制数的加法运算，则需从低位开始依次对每位数进行调整，才能得到正确的运算结果。

8421 码虽可用于十进制数的运算，但由于其数据表示效率低（4 位二进制数只能表示 1 位十进制数），且运算效率也很低（需要对结果进行调整），所以在实际应用中很少使用。

2．定点数

计算机中主要的数值数据表示方法有定点数表示法和浮点数表示法两种；定点数表示法也是浮点数表示法的基础。

所谓定点数表示，是指小数点被固定在数据中某个特定位置上的数据表示方法。实际应用中，通常把小数点固定在数据最低有效位的右边，或最高有效位的左边。如果选择前者，则所有定点数均为整数，称为定点整数；如果选择后者，则所有定点数均为纯小数，称为定点小数，如下所示：

$$\text{定点整数：}\ x_s x_{n-1} x_{n-2} \cdots x_1 x_0$$

$$\text{定点小数：}\ x_s . x_{n-1} x_{n-2} \cdots x_1 x_0$$

其中，x_s 是数的符号位（后面会专门讨论）。

定点数中，小数点的位置可以看作是默认的。因此，在计算机中表示定点数时，小数点不用表示出来，这给数据的表示带来很大的方便。目前，多数通用计算机在选择定点数格式时，都选择定点整数格式，这类计算机称为定点整数机。本书只讨论定点整数。

符号的表示是定点数表示必须要解决的问题。带符号的定点数在计算机中有原码、补码和反码等几种编码表示方法。

1）原码表示法

原码是最直观的机器数表示法，它以 0 表示正号，以 1 表示负号，直接置于数的最左端（即最高位位置）；而数的数字部分与其绝对值一致。

【例 1.4】 ① 若 $x=+1011100$，则 $[x]_{原}=\mathbf{0}1011100$；

② 若 $x=-0010011$，则 $[x]_{原}=\mathbf{1}0010011$。

其中，x 是数的真值，$[x]_{原}$是数的原码表示，其最高位为符号位。

n 位原码（包括符号位）的数据表示范围是

$$-(2^{n-1}-1) \leqslant x \leqslant 2^{n-1}-1$$

例如，$n=8$ 时，8 位原码的数据表示-127～127；$n=16$ 时，16 位原码的数据表示-32767～32767。

由于原码的数字部分与其绝对值一致，所以原码比较适合乘、除运算；运算时，将两

数的数字部分直接相乘或相除，而乘积或商的符号可由两个运算数据的符号经逻辑“异或”得到。但是，原码不适合做加减运算。

2）反码表示法

一个数的反码可通过其原码求得，方法是：正数的反码与其原码一致；负数的反码与其原码符号位相同，数字位按位取反。

【例 1.5】 ① 若 $x=+1011100$，则 $[x]_{反}=[x]_{原}=\mathbf{0}1011100$；

② 若 $x=-0010011$，则 $[x]_{原}=\mathbf{1}0010011$，$[x]_{反}=\mathbf{1}1101100$。

反码一般不用于计算，但可用来作为原码转换为补码时的中间代码。

3）补码表示法

设$|x|<2^n$，则 x 的补码被定义为

$$[x]_{补}=2^n+x \pmod{2^n}$$

其中，n 为所形成的补码的位数（包括符号位）；mod 表示“模”运算，即“相除取余数”运算，2^n 被称为“模数”，mod 2^n 表示“除以 2^n 取余数”。所形成的 n 位补码中，最高位为符号位，数字部分为 n–1 位，x 的实际取值范围是

$$-2^{n-1}\leqslant x\leqslant 2^{n-1}-1$$

例如，$n=8$ 时，8 位补码的数据表示范围是−128～127；$n=16$ 时，16 位补码的数据表示范围是−32768～32767。

【例 1.6】 设 $x=+1001011$，求其 8 位补码$[x]_{补}$。

解：
$$\begin{aligned}[x]_{补}&=2^8+x &\pmod{2^8}\\&=2^8+(+1001011) &\pmod{2^8}\\&=\mathbf{0}1001011 &\pmod{2^8}\end{aligned}$$

其中，最高位上的 **0** 被看作符号位。由本例可知：一个正数的补码与其原码是一致的。

【例 1.7】 设 $x=-1001011$，求其 8 位补码$[x]_{补}$。

解：
$$\begin{aligned}[x]_{补}&=2^8+x &\pmod{2^8}\\&=2^8+(-1001011) &\pmod{2^8}\\&=\mathbf{1}0110101 &\pmod{2^8}\end{aligned}$$

其中，最高位上的 **1** 被看作符号位。由本例可知：一个负数的补码，其符号位为 1。显然，负数的补码与其原码是不同的。

由以上两例可知：与原码和反码不同，补码的符号位（正为 0，负为 1）不是人为规定的，而是在求补码的运算中求出的，实际上就是运算结果的最高有效数字位。因此，在用补码进行加减运算时，符号位可以像数字位一样参加运算，不用单独处理。这给计算机的加减运算带来很大方便。事实上，计算机都是采用补码来做加减运算的。

除了可用上述补码计算公式求取补码外，还有一种形式上的转换方法，可以在原码和补码之间直接转换，即：正数的补码与其原码相同；对负数，保持其原码（或补码）的符号位不变，数字位按位取反，最后加上 1，即可得到对应的补码（或原码）。

【例 1.8】 设 $x=-1001011$，求其 8 位补码$[x]_{补}$。

解： x 为负数，先求其原码 $[x]_{原}=\mathbf{1}1001011$

符号位不变，数字位按位取反 ↓

$$[x]_{反}=\mathbf{1}0110100$$

加 1　　↓

得到 x 的补码 $[x]_{补}=\mathbf{1}0110101$

需要注意的是，由于同样长度的补码和原码的数据表示范围不完全相同（补码在负数方向的表示范围略大），所以上述转换方法只适用于两者数据范围重叠的部分，即补码可表示的绝对值最大的负数没有对应的原码。

补码在乘、除运算方面比原码略为复杂，但也有很好的技术实现补码的乘、除运算。

综上所述，补码最适合计算机中的数值计算，计算机中的带符号定点数实际采用的都是补码表示法。

下面简要说明补码加减运算及其相关问题。

n 位补码加、减运算规则如下：

$$[x]_{补}+[y]_{补}=[x+y]_{补} \quad (\mathrm{mod}\ 2^n) \tag{1.1}$$

$$[x]_{补}-[y]_{补}=[x]_{补}+[-y]_{补}=[x-y]_{补} \quad (\mathrm{mod}\ 2^n) \tag{1.2}$$

可见，两数补码之和、差，等于两数和、差之补码，这说明补码可以直接加、减。此外，式（1.2）表明，补码减法运算可以转换为补码加法运算。事实上，计算机的运算器就是自动把减法转换为加法来运算的。

【例 1.9】 设 $x=+1010110$，$y=-1001001$，按 8 位补码求 $[x+y]_{补}$。

解：首先求出 x 和 y 的补码

$$[x]_{补}=\mathbf{0}1010110 \qquad [y]_{补}=\mathbf{1}0110111$$

按式（1.1），有

$$\begin{array}{rll} & \mathbf{0}1010110 & [x]_{补} \\ + & \mathbf{1}0110111 & [y]_{补} \\ \hline & \boxed{1}\mathbf{0}0001101 & [x+y]_{补} \quad (\mathrm{mod}\ 2^8) \end{array}$$

从运算结果来看，最高位上产生了进位 1，但在 mod 2^8 的作用下，该位不被保留，所以

$$[x+y]_{补}=\mathbf{0}0001101 \quad (\mathrm{mod}\ 2^8)$$

其符号位为 **0**，说明和为正数。

【例 1.10】 设 $x=+1010110$，$y=+1101001$，按 8 位补码求 $[x-y]_{补}$。

解：首先求出 x 和 y 的补码

$$[x]_{补}=\mathbf{0}1010110 \qquad [y]_{补}=\mathbf{0}1101001$$

由式（1.2）可知，要将减法转换为加法，需要求出$[-y]_{补}$

$$[-y]_{补}=\mathbf{1}0010111$$

由此可得

$$\begin{array}{rll} & \mathbf{0}1010110 & [x]_{补} \\ + & \mathbf{1}0010111 & [-y]_{补} \\ \hline & \mathbf{1}1101101 & [x-y]_{补} \quad (\mathrm{mod}\ 2^8) \end{array}$$

所以

$$[x-y]_{补}=\mathbf{1}1101101 \quad (\mathrm{mod}\ 2^8)$$

从运算结果来看，符号位为 1，说明差为负数。

【例 1.11】 设 $x=+1010110$，$y=+1001001$，按 8 位补码求 $[x+y]_{补}$。

解：首先求出 x 和 y 的补码