

普通高校系列教材·信息技术

汇编语言程序设计

普通高校系列教材(信息技术)编委会组编

王 勇 编

学习参考



42

南京大学出版社

普通高校系列教材·信息技术

《汇编语言程序设计》学习参考

王 勇 编

南京大学出版社

内
容
简
介

本书是全国普通高校教材《汇编语言程序设计》的配套辅导教材。在内容的组织安排上,本书根据课程知识结构特点,以及学习的一般规律,将全书内容细分为学习方法指导、习题解答、典型题解与分析、模拟测试卷和实验指导五个部分。可以说这些内容基本上囊括了学习者在学习过程中需要解决的问题,因此该书所提供的辅导较为系统、全面,对学习者有很大帮助。

本书不仅可以作为《汇编语言程序设计》的配套教材,帮助学习者在学习该课程时,更好更快地理解、掌握和运用相关知识,而且也可作为有关人员在学习汇编语言程序设计时,作为参考书使用。

图书在版编目(CIP)数据

《汇编语言程序设计》学习参考/王勇编. —南京:南京大学出版社,2003.5

ISBN 7-305-02788-X

I. 汇... II. 王... III. 汇编语言-程序设计-高等学校-教学参考资料 IV. TP313

中国版本图书馆 CIP 数据核字(2003)第 037902 号

书 名 《汇编语言程序设计》学习参考
编 者 王勇
出版发行 南京大学出版社
社 址 南京市汉口路 22 号 邮编 210093
电 话 025-3596923 025-3592317 传真 025-3303347
网 址 www.njupress.com
电子函件 nupress1@public1.ptt.js.cn
经 销 全国新华书店
印 刷 合肥学苑印刷厂印刷
开 本 787×1092 1/16 印张 11.25 字数 270 千
版 次 2003 年 6 月第 1 版 2003 年 6 月第 1 次印刷
书 号 ISBN 7-305-02788-X/TP·124
定 价 17.00 元

* 版权所有,侵权必究

* 凡购买南大版图书,如有印装质量问题,请与所购
图书销售部门联系调换

普通高校系列教材(信息技术)编委会

主 任:孙钟秀 中国科学院院士

副主任:张福炎 南京大学教授

陈国良 中国科学技术大学教授

钱洲胜 中国计算机函授学院教授

委 员(按姓氏笔画排序):

于学锋 中国计算机函授学院
王文兰 桂林电子工业学院
伍良富 成都电子科技大学
刘存书 郑州信息工程大学
朱宝长 西安电子科技大学
杜象元 上海交通大学
李学干 西安电子科技大学
何淑兰 北京科技大学
张绍林 河北行政学院
张民坤 云南工业大学
赵良全 新疆大学
洪志全 成都理工学院
高 平 浙江大学
曹翊旺 湖南省计算机高等专科学校
韩国强 华南理工大学
葛 燕 中国科学技术大学
谭耀铭 南京大学

王佩珠 西安交通大学
王蔚韬 重庆建筑大学
成松林 东南大学
朱大奇 安徽工业大学
孙德文 上海交通大学
李茂青 厦门大学
杨来利 兰州大学
张凤祥 华中科技大学
张维勇 合肥工业大学
张景书 哈尔滨工程大学
皇甫正贤 东南大学
姚君遗 合肥工业大学
陶世群 山西大学
梁文康 山东大学
舒 洪 南昌大学
解世耀 辽宁大学
黎庆国 合肥工业大学

出版前言

近些年来我国的高等教育事业有了长足的发展,高校招生人数年年递增,越来越多的年轻人有机会接受正规的高等教育。这一举措无疑对我国的社会进步和经济发展有着重要的意义。但是人们也深刻地认识到,高等教育质量的好坏是一个不容忽视的关键性问题,而保证教育质量的一个重要环节就是抓好教材建设。但是教材内容陈旧、教学手段落后的现象一直存在着。尤其像计算机技术这样的新兴领域发展迅猛、知识更新日新月异、教学内容落后于新技术新知识的矛盾显得尤为突出。基于上述两方面考虑,在南京大学出版社的鼎力相助下,一个以组编高校信息、电子类专业教材为主要任务的教材编委会成立了。

针对我国高等教育的现状和信息、电子技术的发展趋势,编委会组织部分高校的专家教授进行了深入的专题研讨。大家一致认为,在当前情况下组编一套紧跟新技术发展、符合高校教学需要、满足大学生求知欲望的系列教材势在必行,这将有助于规范教学体系、更新教学内容、把握教学质量,培养合格人才。专家们还对教材的结构、内容、体例及配套服务等提出了具体要求。为了能使这套教材逐步完善,并促进全国各地高校教学质量的提高,编委会决定在教材之外认真做好三件事:第一,为每本教材配备一本供学生使用的学习参考书,其主体内容为学习方法指导、习题分析与解答、典型题解或课程设计、模拟测试卷及解答、实验指导书;第二,定期对教材内容进行修订,及时补充新技术新知识,并根据具体情况组编新的教材;第三,有计划地组织各地高校教师进行教学交流与研讨,通过这种途径来提高偏远地区的师资水平。我们相信,通过各方面的大力支持和大家的不懈努力,这套教材会逐步被广大师生所接受,并在使用过程中得到完善、充实。

大家都知道,组编这样一套系列教材是个牵涉面很广的大工程。这个工程不仅在起步阶段需要得到各级教育主管部门、各高等院校、出版社的大力支持和协助,而且在使用过程中也离不开各位专家、教授、学生的热心呵护和指导。因此,殷切期待所有的能人志士关心我们,帮助我们,向我们提出好的建议或意见,为我们指出教材中的不足之处。

最后,感谢所有为本套系列教材出版付出辛勤劳动的同志们。

普通高校教材(信息技术)编委会

2003年3月

编者的话

汇编语言程序设计是计算机应用专业的核心课程之一,是学好操作系统、微型机及其应用、计算机控制技术等课程的基础。因此,学好汇编语言程序设计这门课程非常重要。

编写本书的指导思想,是为学习者理清学习思路,抓住课程的重点、难点,并且培养学习者运用所学知识点去分析问题和解决问题的能力。全书分为五个部分,具体内容有:

1. 自学方法指导

掌握正确的学习方法,是取得良好学习效果的根本保障。作为一本学习参考书,最根本的目的,就是要向学习者传授一种好的学习方法。本书自学方法指导部分的内容,体现了作者对该门课程的透彻理解,在此基础上,总结出一套行之有效的学习方法,希望学习者在掌握方法这一利器的基础上进行学习,从而达到事半功倍的效果。

2. 习题解答

在这一部分的内容中,对教材课后练习进行了解答。

3. 典型题解与分析

“典型题”是具有“代表性”的题目,通过对典型题的分析与解答,可以向学习者提供一种解题思路,开阔其视野,从而使他们能够做到举一反三,融会贯通。

4. 模拟测试卷

模拟测试卷部分提供了两套模拟测试卷,并给出了参考答案。希望通过这种模拟测试,让学习者能够对自己的学习情况作出正确评估,找出学习中还存在的问题,从而做到有针对性去解决。

5. 实验指导

汇编语言程序设计的实践性和应用性很强。因此,在这部分内容中提出了多项实验要求,并给出了具体的指导说明。目的是培养学习者的动手能力,能够真正掌握汇编语言程序设计的方法和技巧。

由于时间仓促,编者水平有限,书中存在错误或不足之处在所难免,恳请广大读者批评指正。

编者

2003年3月20日

目 录

第一部分 学习方法指导	(1)
一、课程的性质与要求	(1)
二、怎样学好《汇编语言程序设计》	(2)
三、重点、难点讲解	(3)
第二部分 习题解答	(23)
第一章	(23)
第二章	(24)
第三章	(31)
第四章	(35)
第五章	(41)
第六章	(45)
第七章	(53)
第八章	(62)
第三部分 典型题解与分析	(76)
第四部分 模拟测试卷	(122)
模拟测试卷(一)	(122)
模拟测试卷(一)分析与解答	(127)
模拟测试卷(二)	(130)
模拟测试卷(二)分析与解答	(136)
第五部分 实验指导	(140)
实验一 顺序程序设计	(141)
实验二 分支程序设计	(143)

实验三 循环程序设计	(145)
附录 1 ASCII 代码表	(147)
附录 2 调试程序 DEBUG 的使用	(148)
附录 3 BIOS 中断调用	(158)
附录 4 出错信息	(163)

第一部分 学习方法指导

一、课程的性质与要求

1. 课程的性质

汇编语言程序设计是一门重要的专业技术课程。通过本课程的学习,不仅可获得计算机系统中有关软件方面的基础知识,而且可为将来从事计算机应用打下良好的理论和技术基础。也只有在学好本课程的基础上,才能学好操作系统、高级语言程序设计和计算机控制系统等课程。另外,在本课程的学习过程中,将涉及到数制、二进制运算、逻辑运算、数据的存储等知识。因此,在学习本课程之前,应掌握“数据结构”和“数字逻辑”两门课程中的有关内容。不仅如此,本课程与“微型机及其应用”课程的内容也是密切相关的。本课程中的很多内容都涉及到微型机的硬件环境,特别是 Intel 8086 CPU 的构成和特点,这些内容是“微型机及其应用”课程中所详细介绍的;而有关 8086 的寻址方式、指令系统及程序设计的基本方法和技巧则是本课程所详细介绍的。这就要求在学习“汇编语言程序”与“微型机及其应用”时,要互相兼顾。

2. 本课程的基本要求

通过本课程的学习,读者应掌握有关指令及程序的概念、组成和功能;掌握有关汇编语言语句、源程序的书写格式及规定;熟悉和掌握程序的四种基本结构形式及程序设计方法;具备汇编语言程序的分析能力,以及运用汇编语言设计程序的能力。具体来说,通过学习,读者应达到如下要求:

①熟练掌握有关计算机系统的一些基本概念、基本组成和 8086 汇编语言编程的基本硬件模型组成。

②深刻理解寻址方式、指令系统和指令的概念。熟练掌握 8086 的各种寻址方式的含义和书写格式,熟知所寻找的操作数据的出处或去处。着重理解存储器操作数形成有效地址的各种方法和规定、段地址的约定或超越的规定及书写格式。深刻理解和熟练掌握 8086 指令系统中各种常用指令的操作内容,及指令执行过程中,参加操作的源操作数和结果(目的)操作数的来龙去脉,以及对程序状态寄存器的状态位的影响。会使用各种常用指令分析程序和编写程序。

③了解和掌握各种汇编语言语句的格式和要求,汇编语言源程序的格式和要求;熟悉汇编语言程序上机过程;理解和掌握各类伪指令的助记符、书写格式和用处,及它们对操作数部分的规定;了解宏指令的概念、书写规定和用处。

④深刻理解程序的基本概念,掌握程序设计的基本步骤和基本方法;理解和掌握顺序程序的结构形式和程序设计方法。

⑤熟知各种形式的分支程序结构和各自的结构特点;掌握双分支和三种多分支程序设计的方法。

⑥熟悉循环程序的基本结构形式及各组成部分的内容和功能;熟练掌握单重循环和多重循环程序设计的方法和技巧,以及控制循环的四种设计方法和技巧。

⑦深刻理解子程序的概念、基本结构形式和在程序设计中的作用;熟悉子程序文件说明的重要性及书写子程序文件说明的规定;明白主程序和子程序的调用关系;掌握子程序调用和返回指令在子程序设计中的应用,了解子程序调用和返回过程所完成的操作;理解保护断点、保护现场、恢复现场和恢复断点的意义和方法;分清宏指令和子程序的区别;熟练掌握子程序设计方法和技巧;初步掌握子程序参数传递的方法和子程序嵌套方法;了解 DOS 系统调用的基本内容及调用方法。

⑧掌握各种实用程序在分析问题、确立算法、画出程序流程图、分配存储器和工作单元、选择指令和合适的基本程序结构形式等问题上的设计考虑、编程方法和技巧。

⑨掌握上机练习的两种方法:MASM(ASM)和 DEBUG;熟悉上机练习的四个过程:编辑、汇编、连接和调试;掌握编辑、汇编、连接和调试四个实用程序的使用方法,记住和掌握编辑和调试程序中几个常用软件的使用;能够根据实验要求,设计出汇编语言源程序,并在微机上进行编辑、汇编、连接和调试并修改程序,直到程序无语法和逻辑设计错误,能正常运行,满足设计要求。

二、怎样学好《汇编语言程序设计》

具体说来,学习者在学习中,应注意以下几个问题:

1. 掌握学习方法,树立学习信心

学习的天敌就是害怕困难,丧失信心。其实,要做好任何一件事,都必须克服畏难情绪,树立坚定的信心。在学习过程中,当遇到一些棘手的问题时,只要有足够的信心来面对问题,坚持不懈地努力,耐心地分析问题、剖析问题,最终总能找到解决问题的方法。当找到解决问题的办法时,可能你会发现要解决的问题其实是如此地简单。有了信心,还得讲究学习方法,方法是学习任何东西的利器。在开始学习某一章内容之前,应先了解对该章的学习应该达到怎样的要求,以便在阅读教材时做到心中有数、有的放矢。阅读教材时,可先全面浏览教材,大致了解每个章节的主要内容和特点,即先粗读后细读,做到循序渐进地阅读教材。一般来说,在遇到的疑难问题没有解决以前,不宜学习新的内容。但也不要因为个别不影响学习新内容的细节问题没有解决,而停止不前。在学习过程中,不单要动脑筋思考问题,更需要勤于复习学过的内容,正所谓温故而知新。掌握课程的特点,对学习也有极大的帮助。汇编语言程序设计是建立在指令和基本的程序结构形式和设计方法基础上的,在学习本课程时,应充分重视这两方面内容的学习、理解、记忆和掌握,深入了解和牢记各种指令的助记符,各种寻址方式的规定和书写格式。同时,还应明白参与操作的数据存储单元的地址及数据本身类型、大小,操作结果的存储单元地址;熟悉和记住指令所进行的具体操作;深入了解和掌握四种基本程序的结构形式;熟练掌握四种基本的程序设计方法和技巧;深入了解和掌握对实际问题建立编程算法、画出程序执行流程、选择合适的指令、设置工作单元、编出正确

的源程序等方面的分析问题和解决问题的方法和技巧。

2. 重视对基本概念、基础知识的理解与应用

必须集中精力,吃透教材中每一个知识点,对基本概念必须深刻理解,对基本理论必须彻底搞懂,对基本方法必须牢固掌握,对疑难问题可查阅参考资料。教材中的内容,基本上是按由浅入深、循序渐进的方式安排的。如果前面的内容没有掌握好,后面的内容也就很难掌握。在教学实践过程中,很多同学在学习了一段时间以后,对很多问题不理解,主要原因就是对前面介绍的基本知识和概念没有吃透,由此可见,真正地理解基本知识和概念是非常重要的。因此,建议大家在学习的过程中,一定要重视对基本知识和概念的学习与消化,并且要善于利用这些知识来理解、思考并处理在学习和实践的过程中遇到的有关问题。

3. 多总结归纳,举一反三

在学习过程中,要勤于思考、善于思考,注重对问题的归纳、总结和比较,对容易混淆的知识点,要找出它们之间的异同点,以求对知识点的深刻领会。另外,对于学习不要死啃书本,应做到灵活掌握、融合贯通,最终做到学用结合。

4. 多上机实践

本课程是一门实践性很强的课程。实践有两种含义:一种是要多阅读程序,能分析现成的实用程序,能读懂程序解决的是什么样的实际问题,能明白程序执行的流程,能明确数据操作的情况,能知晓每条指令的具体操作;第二种是亲自编写程序。教材中每章都有大量的习题,可选择做作业。一定要按步骤、按要求、按格式进行编写程序。为了检验编写的程序的正确性,应尽可能找机会上机进行练习。大家只有勤上机进行实际操作,才能取得较好的学习效果。在学习过程中适时上机,通过实际操作可以深化对有关基本知识和概念的理解,使得对所学内容的印象深刻,不易忘记,这样可提高学习效率;同时,通过上机效仿书中操作的实例,还可熟悉操作过程和方法,验证操作结果,从而逐步培养自己的实际动手能力,提高操作技能。

三、重点、难点讲解

第一章 基础知识

本章主要简单介绍了微型计算机硬件系统和软件系统的基本知识。学好本章内容,将学习汇编语言程序设计打下良好的基础。

1. 汇编语言和高级语言程序设计的特点和区别

汇编语言用英文单词助记符表示机器指令中的操作码,使汇编指令非常直观,克服了机器语言用数码表示操作和地址,数码含义不直观、难以记忆、编写的程序也难以阅读和调试的缺点。但用汇编语言编写的源程序不能被机器直接识别,必须经汇编程序翻译成等价的机器语言程序(或称目标程序)后,才能为机器所识别。这种用汇编程序将源程序翻译成目

标程序代码的过程,称为汇编。

汇编语言是面向机器的,它的执行速度与机器指令接近;同时用汇编语言编写的源程序比编译程序要小得多,所以占内存较少,高级语言不能有效地产生机器语言程序,而汇编语言恰好能弥补高级语言的不足,它的指令语句与机器指令一一对应,编写的程序较为精练,使用效率较高。

高级语言是一种算法语言,易学易懂,使用方便,通用性强,便于推广与交流,且它有文件操作功能,可以将有关数据以文件方式存入外存储器中。但高级语言程序不能直接执行,需借助编译(翻译)程序,将其翻译成机器语言程序,才能被机器执行。由于高级语言是面向过程或对象的,主要考虑怎样容易编写程序,因此在编译生成目标代码时,不可避免地出现编译后的目标程序不精练、过于冗长、执行速度慢等特点。

2. 8086/8088 CPU 寄存器的分类和各寄存器的主要作用

8086/8088 共有 14 个 16 位寄存器,按照其功能可以分为以下四组:

(1) 通用寄存器组

通用数据寄存器 AX、BX、CX、DX 为 4 个 16 位寄存器,也可以作为 8 个 8 位寄存器来使用。

(2) 指针及变址寄存器

指针及变址寄存器包括 SP、BP、SI 和 DI 四个 16 位寄存器。堆栈指针寄存器 SP 用于在进行堆栈操作时,确定堆栈区在内存中的位置。但 SP 必须与堆栈段寄存器 SS 一起才能确定当前堆栈操作的物理地址。基址指针寄存器 BP、源变址寄存器 SI 和目的变址寄存器 DI 主要用于扩充寻址方式,和 BX 寄存器一样,用于对操作数据的间接寻址或变址寻址。

(3) 段寄存器组

段寄存器组由代码段(CS)、数据段(DS)、堆栈段(SS)和附加段(ES)四个 16 位寄存器组成。它们的主要用途如下:

8086/8088 在执行取指令操作、跳转执行或寻找存储器操作数地址时,采用了分段寻址方式,在同一时刻可将内存分为四个逻辑段,段首地址由段寄存器的值决定。段寄存器值为 16 位二进制数,称为段地址。一条指令或操作数据的物理地址,是由段地址和偏移地址共同确定的。采用段地址的方法能使 8088 在 1MB 的范围内对内存进行寻址。

(4) 控制寄存器组

控制寄存器组由指令指针寄存器(IP)和状态标志寄存器(PSW)组成。IP 寄存器供系统使用,其内容为下一条将要执行的指令的偏移地址。IP 寄存器应与代码段寄存器 CS 相配合,才能形成操作指令的物理地址。PSW 寄存器用以反映系统状态和运算操作结果的特征。8086/8088 PSW 寄存器占 2 个字节,共有 9 个标志位,具体情况如图 1-1 所示。

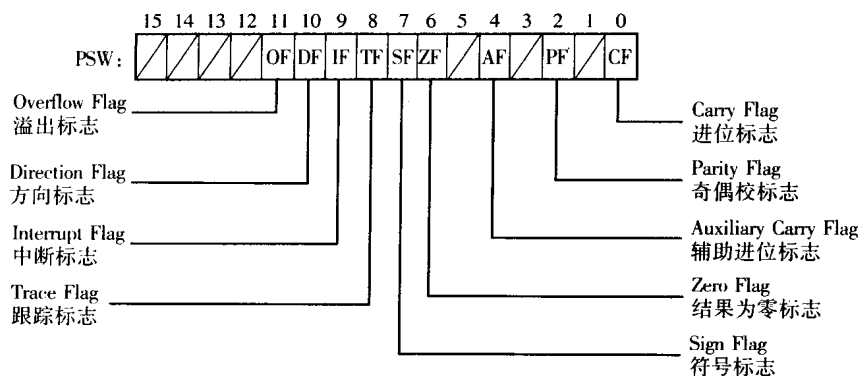


图 1-1 PSW 程序状态标志寄存器

3. 状态寄存器 PSW 中所设置的 9 种状态标志产生的条件

(1) OF 标志产生的条件

在运算过程中,当运算结果超出了机器能表示的数值范围,则 $OF = 1$,表示产生溢出,否则, $OF = 0$,表示未产生溢出。

(2) SF 标志产生的条件

当运算结果为负数时, $SF = 0$,反之, $SF = 1$ 。

(3) ZF 标志产生的条件

当运算结果为零时, $ZF = 1$,反之, $ZF = 0$ 。

(4) CF 标志产生的条件

当运算过程中发生进位或借位时, $CF = 1$,否则, $CF = 0$ 。

(5) AF 标志产生的条件

在运算过程中,当第三位有进位或借位时, $AF = 1$,否则, $AF = 0$ 。

(6) PF 标志产生的条件

执行指令后,产生的结果操作数,若其低 8 位中“1”的个数为偶数时, $PF = 1$,否则, $PF = 0$ 。

(7) DF 标志产生的条件

当需要 SI 和 DI 执行减量操作时,在程序中置 $DF = 1$,反之,置 $DF = 0$ 。

(8) IF 标志产生的条件

在程序中,当允许中断时,置 $IF = 1$,否则,置 $IF = 0$ 。

(9) TF 标志产生的条件

在调试程序过程中,当置 $TF = 1$ 时,允许程序在执行一条指令后,产生一次内部中断,以便检查该条指令的执行结果,反之,则置 $TF = 0$ 。

4. IBM-PC 机存储器的组织形式和特点

8086/8088 有 20 条地址线,其直接寻址范围为 1MB,地址编号为 $00000H \sim FFFFFH$ 。但是 CPU 内部的算术逻辑运算单元 ALU 只能进行 16 位运算,因而地址的运算也只能是 16 位的,从而也就决定了 8086/8088 的最大范围只能是 64KB 空间。为了能完成对 1MB 空间的遍访寻址,系统设置了四个段寄存器,将 1MB 空间分成若干个 64KB 段,然后通过如图 1-2 所示的方法形成 20 位物理地址。

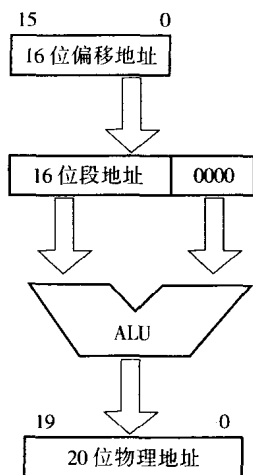


图 1-2 20 位物理地址形成方法示意图

形成 20 位物理地址的具体方法如下:

①当执行取指令操作时,系统自动选择代码段寄存器 CS,再加上由指令指针寄存器 IP 所决定的 16 位偏移量,计算形成所要取的指令的物理地址。

②当执行堆栈操作时,系统自动选择堆栈段寄存器 SS,再加上由堆栈指针寄存器 SP 所决定的 16 位偏移量,计算形成堆栈操作所需要的 20 位物理地址。

③当执行存储器操作数处理指令时,系统自动选择数据段寄存器 DS 或者附加段寄存器 ES,将其内容再加上 16 位地址偏移量,计算形成操作数据的 20 位物理地址。

采用存储器分段的方法,能方便地对堆栈区、程序区、数据区进行隔离,也能满足同一个程序能在内存的不同区域运行,而不需要改变程序本身。

第二章 8086/8088 的寻址方式和指令系统

本章主要介绍了指令的七种寻址方式,以及在不同寻址方式下,源操作数和目的操作数物理地址的确定。掌握指令的定义和组成、操作码和操作数在指令中的作用十分必要。学习指令系统的重点是掌握好寻址方式,以及相关指令执行后对状态寄存器 PSW 各标志位的影响。

1. 寻址方式

寻址方式是指寻找指令中操作数所在物理地址的方法,8086/8088 汇编语言的寻址方式分为对操作数据的寻址方式和对转移地址的寻址方式。

(1) 对操作数据的寻址方式

8086/8088 对数据的基本寻址方式有六种:

①立即数寻址(Immediate Addressing)。该方式所提供的操作数直接出现在指令中,位于操作码后面,与操作码一起放在代码段区域中。立即数寻址方式主要用来给寄存器或内存单元赋初值。

②直接寻址(Direct Addressing)。该方式在指令中给出操作数地址的 16 位偏移量,它与操作码一起存放在代码段区域。操作数一般在数据段区域中,其物理地址为数据段寄存器 DS 中存放的段地址 $\times 10H$ 加上 16 位地址偏移量。该寻址方式是以数据段的地址为基础,故可在多达 64KB 的范围内寻找操作数。对于寻找的操作数来说,还允许操作数在以代码段、堆栈段或附加段为基准的区域中,即只要在指令中指明是段超越的,则 16 位地址偏移量可以和 DS 或 SS 或 ES 相加,形成操作数的物理地址。

③寄存器寻址(Register Addressing)。该方式所提供的操作数在 CPU 的寄存器中。该方式的 操作数可以存放在 CPU 内部的任意一个寄存器中,但应尽量优先选用累加器作目的寄存器,这样可节省指令的执行时间。

④寄存器间接寻址(Register Indirect Addressing)。该方式所提供的操作数在存储单元中,其操作数地址的 16 位偏移量包含在 SI、DI、BX、BP 四个寄存器的任一个寄存器之一中。若以 SI、DI、BX 间接寻址,则通常操作数在现行数据段区域中,即用数据段寄存器 DS 加上 SI、DI、BX 中的 16 位偏移地址作为操作数的物理地址;若以寄存器 BP 间接寻址,则操作数在堆栈区域中,即用堆栈段寄存器 SS 和 BP 中的偏移地址相加作为操作数的物理地址;若指令中规定是段超越的,则 BP 也可以和其他的段寄存器相加,形成操作数的物理地址。

⑤变址寻址(Index Addressing)。该方式以一个段寄存器为地址基准,并由指定的变址寄存器的内容加上指令中给定的 8 位或 16 位偏移量,形成操作数的物理地址。作为寄存器间接寻址的四个寄存器 SI、DI、BX 和 BP 又可用来作为变址寄存器。若用 SI、DI 和 BX 作为变址寄存器,则和数据段寄存器一起形成操作数的物理地址;若用 BP 作为变址寄存器,则与堆栈段寄存器一起形成操作数的物理地址;若指令中指定是段超越的,则可用超越说明的段寄存器作为地址基准。

⑥基址变址相对寻址。该方式是将基址寄存器 BX、BP 和变址寄存器 SI、DI 两种寻址方式组合起来形成一种新的寻址方式。即将基址寄存器(BX 或 BP)的内容加上变址寄存器(SI 或 DI)的内容,再加上指令中指定的 8 位或 16 位偏移量,同样再以一个段寄存器作为地址基准,形成操作数的物理地址。在一般情况下,由 BX、BP 基址寄存器决定所在段寄存器,若用 BX 作为基址,则操作数在数据段区域中;若用 BP 作为基址,则操作数在堆栈段区域中。若在指令中规定段超越,则可用其他规定的段寄存器作为地址基准。

(2) 对转移地址的寻址方式

按控制转移的方式分,转移既可以是有条件转移,也可以是无条件转移;按转移的范围分,既可以是段内转移,也可以是段间转移。有条件转移指令均为段内转移,而无条件转移指令可分别实现段内或段间转移。

①段内转移。段内转移指在同一代码段的范围之内进行转移。此时只需改变 IP 寄存器的内容,即用新的转移目标地址替代原有的 IP 值就可达到转移的目的,代码段寄存器 CS 的内容不变。段内转移可分为段内直接短转移(段内直接寻址)、段内直接近转移(段内直接寻址)、段内间接转移(段内间接寻址)三种形式。

- 段内直接短转移。段内直接短转移的转移地址位移量由符号地址确定,执行转移指令前 IP 的值应是 JMP 指令的下一条指令的地址。位移量为 8 位有符号数,故允许在 $-128 \sim +127$ 字节的范围内转移。

- 段内直接近转移。段内直接近转移和段内直接短转移一样,均采用相对寻址方式,所不同的是段内直接近转移的地址位移量为 16 位有符号数,故其转移范围为 $-32768 \sim +32767$ 字节,即该种转移方式可跳转到当前代码段内的任一个地址。

- 段内间接转移。段内间接转移的寻址方式,可以是除立即数以外的任一种寻址方式。当寻址方式为寄存器时,即将 16 位寄存器的内容送 IP;当寻址方式为指定的存储器字单元时,即将该存储单元的内容送 IP。

②段间转移。段间转移是指从当前代码段转移到另一个代码段去执行程序。此时不仅要修改 IP 寄存器中的偏移地址,而且还要修改代码段寄存器 CS 中的段地址,才能达到目的。此时的转移目标地址应由新的段地址和偏移地址两部分组成。段间转移可分为段间直接转移和段间间接转移两种形式。

- 段间直接转移。段间直接转移使用的是直接寻址方式,指令中转移地址为要跳转到另一个代码段内的偏移地址,汇编程序遇到该指令即将该指令生成两个字的操作数,第一个字操作数的内容为偏移地址 IP 的新值;第二个字操作数的内容为段地址 CS 的新值。

- 段间间接转移。段间间接转移只能通过存储器实现转移,根据存储器间接寻址方式求出有效地址后,将指定存储单元第一个字的内容作为目的偏移地址送 IP,并将下一个字的内容作为目的段地址送 CS,从而实现段间转移。

2. 数据传输指令

数据传送指令包括通用数据传送指令 MOV,地址传送指令 LEA、LDS、LES,进栈和出栈操作指令 PUSH、POP,输入输出指令 IN、OUT,换码指令 XLAT 五类。

3. 算术运算指令

算术运算指令的功能是进行算术运算,包括加法运算指令 ADD、ADC、INC,减法运算指令 SUB、SBB、DEC、NEG、CMP,乘法运算指令 MUL、IMUL,除法运算指令 DIV、IDIV,符号扩展指令 CBW、CWD,十进制运算调整指令 DAA、DAS、AAA、AAS、AAM、AAD 六类。

4. 逻辑运算和移位指令

逻辑运算和移位指令包括逻辑运算指令 AND、OR、NOT、XOR、TEST, 移位操作指令 SHL、SHR、SAL、SAR, 循环移位指令 ROL、ROR、RCL、RCR 三类。

5. 串处理指令

串处理指令包括串传送指令 MOVSB/MOVSX, 串比较指令 COMSB/COMSW, 串查询指令 SCASB/SCASW, 从串取指令 LODSB/LODSW, 存入串指令 STOSB/STOSW 五类。

6. 控制转移指令

控制转移指令可分为如下几类:

(1) 无条件转移指令

无条件转移指令包括段内直接短转移指令 JMP SHORT OPRD, 段内直接近转移指令 JMP NEAR PTR OPRD, 段内间接转移 JMP WORD PTR OPRD, 段间直接转移指令 JMP FAR PTR OPRD, 段间间接转移指令 JMP DWORD PTR OPRD 五种。

(2) 根据单个标志位的条件转移指令

根据单个标志位的条件转移指令包括有进(借)位/无进(借)位时转移指令 JC/JNC, 结果为零/不为零时转移指令 JZ/JNZ, 结果为负/为正时转移指令 JS/JNS, 结果溢出/不溢出时转移指令 JO/JNO, 结果为偶校验/奇校验时转移指令 JP/JNP 五种。

(3) 根据无符号数比较的结果转移指令

根据无符号数比较的结果转移指令有大于转移指令(即 $CF = 0$ 且 $ZF = 0$) JA、大于等于转移指令(即 $CF = 0$ 或 $ZF = 1$) JAE、小于转移指令(即 $CF = 1$ 且 $ZF = 0$) JB 和小于等于转移指令(即 $CF = 1$ 或 $ZF = 1$) JBE 四种。

(4) 根据有符号数比较的结果转移指令

根据有符号数比较的结果转移指令有大于转移指令(即 SF 和 OF 同号且 $ZF = 0$) JG、大于等于转移指令(即 SF 和 OF 同号或 $ZF = 1$) JGE、小于转移指令(即 SF 和 OF 异号且 $ZF = 0$) JL 和小于等于转移指令(即 SF 和 OF 异号或 $ZF = 1$) JLE 四种。

(5) 根据测试 $CX = 0$ 为条件实现转移指令

此类指令主要有 JCXZ OPRD。

(6) 循环指令

循环指令有循环至 $CX = 0$ 止指令 LOOP OPRD、循环至 $CX = 0$ 或 $ZF = 0$ 止指令 LOOPZ OPRD 和循环至 $CX = 0$ 或 $ZF = 1$ 止指令 LOOPNZ OPRD 三种。