

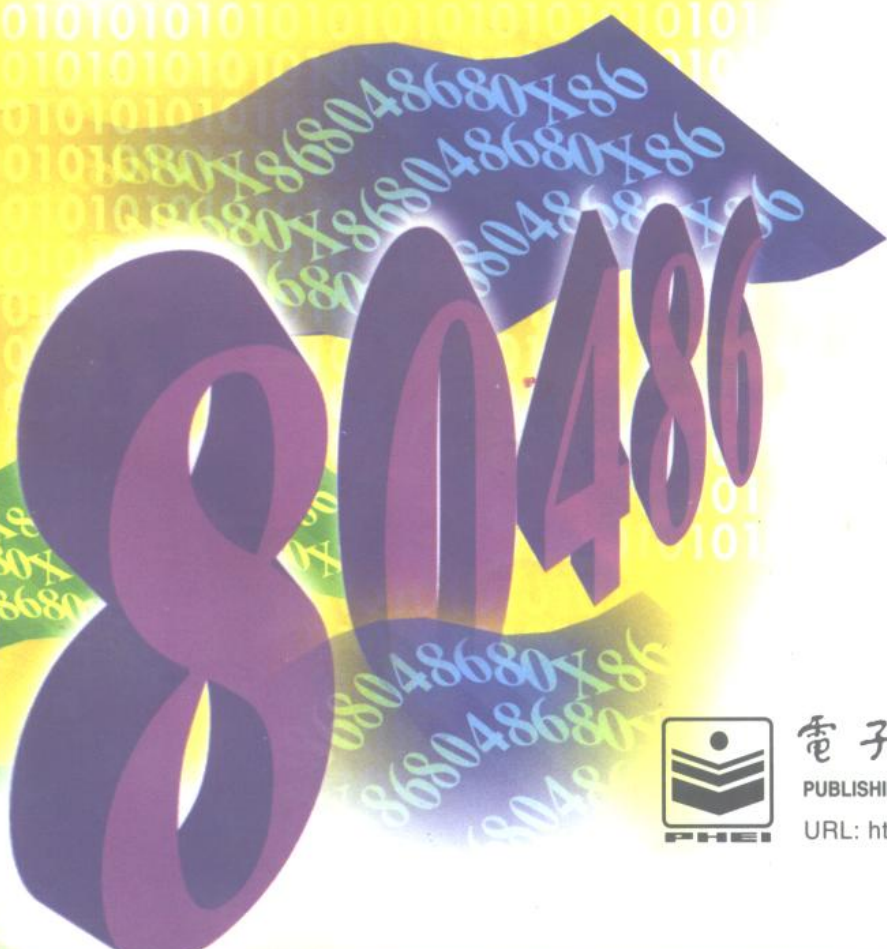


高等专科学校教材

中国计算机学会大专教育学会推荐出版

# 80486(80X86) 汇编语言程序 设计

周学毛 李明钧 赵欢 田华荣 编著



3  
M/1



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

URL: <http://www.phei.co.cn>

7/21  
80X86/1

高等专科学校教材

# 80486(80X86) 汇编语言程序设计

周学毛 李明钧 赵欢 田华荣 编著

电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

## 内 容 简 介

本书以 80486(80X86)微机为背景,较全面地讲述汇编语言程序设计的基本概念、基本原理、基本方法和常用技术,强调以汇编语言程序设计为中心。

本书内容通俗易懂,简明扼要;例题实用,难点分散;程序设计技术先进,容易掌握。适于作大专计算机及相关专业教材,亦适合作非计算机专业本科教材及自学计算机的读者使用。

JS35/27

丛 书 名:高等专科学校教材

书 名:80486(80X86)汇编语言程序设计

编 著:周学毛 李明钧 赵欢 田华荣

责任编辑:张凤鹏

特约编辑:袁 英

排版制作:电子工业出版社计算机排版室

印 刷 者:北京大中印刷厂

出版发行:电子工业出版社出版、发行 URL:<http://www.phei.co.cn>

北京市海淀区万寿路 173 信箱 邮编 100036 发行部电话:68214070

经 销:各地新华书店经销

开 本:787×1092 1/16 印张:14.25 字数:364.8 千字

版 次:1997 年 7 月第 1 版 1997 年 7 月第 1 次印刷

书 号: ISBN 7-5053-4059-X  
G·334

定 价:16.50 元

凡购买电子工业出版社的图书,如有缺页、倒页、脱页者,本社发行部负责调换  
版权所有·翻印必究

## 出版说明

根据国务院关于高等学校教材工作的有关规定,在电子工业部教材办的组织与指导下,按照教材建设适应“三个面向”的需要和贯彻国家教委关于“以全面提高教材质量水平为中心、保证重点教材,保持教材相对稳定,适当扩大教材品种,逐步完善教材配套”的精神,大专计算机专业教材编审委员会与中国计算机学会教育专业委员会大专教育学会密切合作,于1986~1995年先后完成了两轮大专计算机专业教材的编审与出版工作,共出版教材48种,从而较好地解决了全国高等学校大专层次计算机专业教材需求问题。

为及时使教材内容更适应计算机科学与技术飞速发展的需要以及在管理上适应国家实施“双休日”后的教学安排;在速度上适应市场经济发展形势的需要,在电子工业部教材办的指导下,大专计算机专业教材编委会、中国计算机学会大专教育学会与电子工业出版社密切合作,从1994年7月起经过两年的努力制定了1996~2000年大专计算机专业教材编审出版规划。

本书就是规划中配套教材之一。

这批书稿都是通过教学实践,从师生反映较好的讲义中经学校选报,编委会评选择优推荐或认真遴选主编人,进行约编的。广大编审者,编委和出版社编辑为确保教材质量和如期出版,作出了不懈的努力。

限于水平和经验,编审与出版工作中的缺点和不足在所难免,望使用学校和广大师生提出批评建议。

中国计算机学会教育委员会大专教育学会  
电子工业出版社

## 前 言

汇编语言程序设计是计算机专业学生必修的一门核心专业基础课程,对训练学生掌握程序设计方法、学好后续专业课程、从事软件开发和应用有重要作用。计算机技术发展日新月异,本书以 80486(80X86)微机为背景,根据大专计算机“九五”教材编写计划编写。

全书共分八章,以实模式为主体模式,较全面讲述 80486 汇编语言程序设计,包括作为基础的 8086 汇编语言程序设计,读者可自然延伸至 80X86 汇编语言程序设计。第一、二章全面介绍汇编语言程序设计必要的基础知识,主要包括计算机的系统结构、指令系统和程序结构。第三、四、五、六章全面介绍程序设计基本方法,包括三种基本结构程序设计及子程序设计。第七、八章介绍应用程序设计方法。同时提供 80X86 基本指令、上机实习方法等四个附录供读者参考。

本书难点分散,语言通俗简明,只要求读者具有一门高级语言程序设计知识基础。除适合于作为计算机及相关专业专科教材外,亦可作为非计算机专业本科教材,同时也可供自学计算机的读者使用。

由于受时间和篇幅限制,编写大纲中一些想法在本次编写中没能实现,期待能在修订时全部写入。也是由于同样的原因,本书没有讲述 80X87 程序设计,相信读者有了本书基础,参考其他书籍可自学掌握。

本书由周学毛、李明钧、赵欢、田华荣老师共同编写,余明辉、易卫老师参加了部分工作,田华荣、周学毛老师承担编写组织工作。

本书的编写得到了全国计算机大专教育学会专家的全力支持,由学会主任王文章教授主审,在此向为本书编写给予支持和帮助的专家、友人及参考文献的作者表示衷心感谢。

编者

1996 年 7 月于岳麓山

# 目 录

<b>第一章 程序设计基础</b> .....	(1)
第一节 汇编语言程序设计.....	(1)
第二节 数据表示方法.....	(2)
第三节 计算机系统结构.....	(6)
第四节 指令格式和寻址方式.....	(16)
第五节 80486 指令系统.....	(19)
习题一.....	(23)
<b>第二章 汇编语言程序</b> .....	(24)
第一节 汇编语言程序结构.....	(24)
第二节 基本语法成分.....	(26)
第三节 汇编语言语句.....	(30)
第四节 伪指令.....	(30)
第五节 程序设计方法.....	(37)
习题二.....	(38)
<b>第三章 简单程序设计</b> .....	(39)
第一节 算术运算指令.....	(39)
第二节 位处理运算指令.....	(45)
第三节 输入/输出 DOS 功能调用.....	(47)
第四节 简单程序设计.....	(51)
习题三.....	(55)
<b>第四章 分支程序设计</b> .....	(57)
第一节 分支程序的结构.....	(57)
第二节 转移指令.....	(57)
第三节 分支程序设计方法.....	(60)
第四节 分支程序设计举例.....	(68)
习题四.....	(75)
<b>第五章 循环程序设计</b> .....	(77)
第一节 循环控制指令.....	(77)
第二节 串操作与重复前缀指令.....	(79)
第三节 循环程序设计方法.....	(82)
第四节 循环程序设计举例.....	(93)
习题五.....	(109)
<b>第六章 子程序设计</b> .....	(110)
第一节 子程序设计基础.....	(110)

第二节	子程序设计方法	(112)
第三节	递归子程序	(122)
第四节	子程序设计举例	(124)
习题六		(134)
<b>第七章</b>	<b>输入/输出和中断程序设计</b>	<b>(136)</b>
第一节	I/O 设备数据传送方式	(137)
第二节	程序直接控制方式	(140)
第三节	中断和中断处理程序设计	(143)
第四节	BIOS 中断调用	(151)
第五节	磁盘文件管理	(159)
习题七		(171)
<b>第八章</b>	<b>程序设计综合实例</b>	<b>(173)</b>
实例一	彩色图形显示	(173)
实例二	扬声器驱动与音乐编程	(178)
实例三	异步串行通讯	(183)
<b>附录一</b>	<b>80X86 指令一览</b>	<b>(192)</b>
<b>附录二</b>	<b>上机实习方法</b>	<b>(210)</b>
<b>附录三</b>	<b>汇编出错信息一览</b>	<b>(212)</b>
<b>附录四</b>	<b>DEBUG 命令一览</b>	<b>(216)</b>
<b>参考文献</b>		<b>(218)</b>

# 第一章 程序设计基础

本章作为开篇,主要介绍学习汇编语言程序设计所必需的计算机软硬基础知识、计算机的系统结构、指令系统。

## 第一节 汇编语言程序设计

### 一、程序设计语言与程序设计

#### (一)指令与程序

指令是用于指出计算机要进行的操作和操作对象的一组代码,实际上是计算机能识别的一组二进制代码。

一台计算机全部指令的集合,构成该计算机的指令系统。指令系统是计算机基本功能的体现,不同的计算机指令系统不同。

程序是为使计算机完成工作、实现预期目的而用程序设计语言设计的一系列操作,其最终形式是指令序列。

#### (二)程序设计语言

程序设计语言是程序设计人员和计算机进行会话的语言,它遵循一定的规则和形式,构成程序的实现工具。

设计程序设计语言时有一基本前提:既让程序设计人员感到方便,又让计算机最终能识别。二者之间相去甚远,至今尚未得到理想的解决,兼顾二者设计了多种程序设计语言,有的语言接近机器,有的语言接近人。

##### 1. 面向机器的机器语言和汇编语言

机器语言是以计算机硬件能直接执行和理解的指令系统为基础而形成的语言,它为特定的计算机面设计。相应机器语言编写的程序称为机器语言程序,机器语言编写的程序质量高、运行速度快、占用资源少,但不易阅读理解、编写难、调试难、通用性差。除用于编写机器的核心系统程序外,在实际中很少直接使用。

汇编语言是一种符号化了的机器语言,即用助记符号代替机器语言的二进制代码。汇编语言的设计在一定程度上克服了机器语言的不足,同时保留了机器语言的长处。

##### 2. 面向过程的高级语言

为了更加易学易用,提高程序的通用性,设计了大量接近自然语言的程序设计语言。这些语言面向用计算机求解问题的过程,不依赖具体机器,与特定机器相分离,采用接近自然语言的词汇。典型的高级语言有 BASIC、PASCAL、C、FORTRAN、COBOL 语言等。

##### 3. 面向对象的高级语言

程序设计语言的更高形式是面向求解问题本身的高级语言。典型的面向对象的语言有 C++、Smalltalk、面向对象的 PASCAL 语言等。



### (三)算法与程序设计

设计程序设计语言的目的是用来编制程序,通过计算机对程序的执行来求解问题。

程序设计的关键是设计算法。所谓算法就是求解问题的有限步骤,具有有穷性、确定性、能行性、输入、输出五大特征。

## 二、汇编语言与汇编语言程序设计

### (一)汇编语言与宏汇编语言

汇编语言是引进了助记符的机器语言,其语句基本上对应机器语言的指令,通常专属于某种机型或某一系列机型,能将计算机的功能全部提供给程序设计者。但也要求程序设计者在计算机内部元件上操作,对机器性能有更深入的了解。学汇编语言要了解机器的硬件、指令系统、寻址方式、汇编语法等。

汇编语言一般用在系统软件的开发、扩充或修改系统设备及对速度、空间有特别要求的环境。

实际使用的往往是宏汇编语言,它包含一般汇编语言的功能,采用了高级语言的一些特性,是一种接近高级语言的汇编语言,仍属于汇编语言的范畴。

汇编语言及汇编语言程序设计是计算机专业的一门核心专业基础课,不同的机器配备的汇编语言不同。本教材介绍的是广泛应用的PC系列微机的80486(80X86)汇编语言与汇编语言程序设计。

### (二)汇编程序与汇编

用汇编语言编写的程序称为汇编语言源程序,简称源程序,在DOS下的扩展名为ASM。源程序在计算机中不能直接运行,必须通过一个称为汇编程序的机器语言程序将源程序翻译成相应的目标程序才能运行。

源程序通过汇编程序翻译成目标程序的过程称为汇编。

汇编形成的目标程序其实也不能直接运行,还必须经过连接形成可执行的代码程序才能运行。

### (三)汇编语言程序设计

汇编语言程序设计是用汇编语言设计汇编语言程序的过程,实际上是用汇编语言去表现和实现计算机完成的工作。包括对操作进行描述与对数据进行描述两个方面,一般采用结构化程序设计方法。

学程序设计语言不是目的,只是手段,学程序设计才是目的。

## 第二节 数据表示方法

### 一、二进制及十六进制数

计算机作为信息处理工具,数据表示问题是计算机工作的出发点,计算机内部采用二进制作为数据表示的基础,其他表示方法由二进制派生出来。

### (一)二进制数(Binary)

二进制数由 0、1 二个数码构成,基数为 2,第  $i$  位的权为  $2^i$ ,运算逢二进一。

书写时在尾部加注字母 B 或下标 2 描述。

二进制数  $a_n a_{n-1} \dots a_0 . b_{-1} b_{-2} \dots b_{-m}$  的值是:

$a_n \times 2^n + a_{n-1} \times 2^{n-1} \dots + a_0 + b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + \dots + b_{-m} \times 2^{-m}$ , 其中  $a_i$  和  $b_i$  为 0、1 两个数码中一个。

例如:  $101011B = 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 43_{10}$

$n$  位二进制数可以表示  $2^n$  种组合。4 位能表示 16 种组合、0 至 15 的整数,8 位能表示 256 种组合、0 至 255 的整数,16 位能表示  $64 \times 1024(64K)$  种组合。

### (二)十六进制数(Hexadecimal)

使用二进制数在计算机中容易实现,便于存储,抗干扰性强,但对人的阅读、书写、记忆、输入等不方便,尤其位数较多时。

若干年前,程序员发现操作一般是对位的“组”,而不是个别的一些位操作。最早的微处理器 4 位,自然想到用 4 位一组缩写二进制数,即十六进制数。

十六进制的数由 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F 十六个数码构成,其中数码 A、B、C、D、E、F 对应于十进制数 10、11、12、13、14、15,基数为 16,第  $i$  位的权为  $16^i$ ,运算逢十六进一。

书写时在尾部加注字母 H 或下标 16。描述以字母开始的十六进制数的前面应加一数码 0。其他进制数的书写加注相应的字母或下标,缺省一般指十进制数。

十六进制数  $a_n a_{n-1} \dots a_0 . b_{-1} b_{-2} \dots b_{-m}$  的值是:

$a_n \times 16^n + a_{n-1} \times 16^{n-1} \dots + a_0 + b_{-1} \times 16^{-1} + b_{-2} \times 16^{-2} + \dots + b_{-m} \times 16^{-m}$ , 其中  $a_i$  和  $b_i$  为 0、1、...、F 十六个数码中一个。

例如:  $14AFH = 1 \times 16^3 + 4 \times 16^2 + 10 \times 16 + 15 = 5295_{10}$

汇编语言调试系统的数全部用十六进制数,汇编列表文件中代码指令及内存地址也用十六进制数表示。十六进制数是汇编语言的书写工具,应熟悉它。

### (三)八进制数(Octal)

计算机中一个字节是 8 位,也常用八进制数来缩写二进制数。

八进制数由 0、1、2、3、4、5、6、7 八个数码构成,基数为 8,第  $i$  位的权为  $8^i$ ,运算逢八进一。

八进制数  $a_n a_{n-1} \dots a_0 . b_{-1} b_{-2} \dots b_{-m}$  的值是:

$a_n \times 8^n + a_{n-1} \times 8^{n-1} \dots + a_0 + b_{-1} \times 8^{-1} + b_{-2} \times 8^{-2} + \dots + b_{-m} \times 8^{-m}$ , 其中  $a_i$  和  $b_i$  为 0、1、...、7 八个数码中一个。

由于字母 O 与数字 0 容易混淆,八进制亦常用尾标 Q 标识。

### (四)数制转换

#### 1. 非十进制数转换成十进制数

采用按权相加法,只需将各位数码与其对应权值的乘积相加即可。

**例 1-1**

$$101101.01B = 1 * 2^5 + 1 * 2^3 + 1 * 2^2 + 1 + 1 * 2^{-2} = 45.25_{10}$$

$$345Q = 3 * 8^2 + 4 * 8^1 + 5 * 8^0 = 229_{10}$$

$$0F2DH = 15 * 16^2 + 2 * 16^1 + 13 * 16^0 = 3885_{10}$$

**2. 十进制数转换成非十进制数**

转换时将整数部分与小数部分分别转换。

整数部分采用除基数取余法，直至商为 0。先得到的余数为低位，后得到的余数为高位。

小数部分采用乘基数取整法，直至乘积为整数或达到控制精度。

**例 1-2**

$$57_{10} = 111001B = 71Q = 39H$$

$$\begin{array}{r} 2 \overline{) 57} \\ \underline{28} \quad \dots 1 \\ 2 \overline{) 28} \\ \underline{14} \quad \dots 0 \\ 2 \overline{) 14} \\ \underline{7} \quad \dots 0 \\ 2 \overline{) 7} \\ \underline{3} \quad \dots 1 \\ 2 \overline{) 3} \\ \underline{1} \quad \dots 1 \\ 0 \quad \dots 1 \end{array}$$

$$\begin{array}{r} 8 \overline{) 57} \\ \underline{7} \quad \dots 1 \\ 0 \quad \dots 7 \end{array}$$

$$\begin{array}{r} 16 \overline{) 57} \\ \underline{3} \quad \dots 9 \\ 0 \quad \dots 3 \end{array}$$

$$0.625_{10} = 0.101B = 0.5Q = 0.AH$$

$$0.625 * 2 = 1.25 \dots 1 \quad 0.625 * 8 = 5 \dots 5 \quad 0.625 * 16 = 10 \dots A$$

$$0.25 * 2 = 0.5 \dots 0$$

$$0.5 * 2 = 1 \dots 1$$

**3. 二、八、十六进制数之间的转换**

将二进制数转换成八进制数可按三位一组进行，转换成十六进制可按四位一组进行，每一组对应八进制或十六进制相应数码。分组时如位不够，整数部分在最左边补 0，小数部分在最右边补 0，参见表 1-1。

将八进制数转换成二进制数，只需将八进制数一位对应换成二进制数三位即可。同样对十六进制，只需将其一位对应换成二进制数四位即可，参见表 1-1。

表 1-1 十、二、八、十六进制对应关系表

十进制	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
二进制	000	001	010	011	100	101	110	111								
八进制	0	1	2	3	4	5	6	7								
二进制	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
十六进制	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

**例 1-3**

$$1100100.11010B = \underline{001} \underline{100} \underline{100}. \underline{110} \underline{100} = 144.64Q$$

$$= 0110\ 0100.1101\ 0000 = 64.DH$$

反过来即是八进制数 144.64,十六进制 64.D 转换所得的二进制数。

## 二、内存中的数据

数的原值称为真值,它用常规的数符正负号及常规的数码表示。

在计算机中,数的符号也用二进制来表示。连数符一起数字化了的数,称为机器数。一般用最高有效位表示数的符号,正数用 0 表示,负数用 1 表示。机器数可以用不同码制表示,常用的有原码、补码、反码三种表示法,广泛应用的原码和补码两种表示法。

### (一) 数的原码表示

原码表示是一种比较直观的机器数表示方法,原码与真值的区别仅仅是符号位数字化。

+1011 采用原码表示为 01011,-1011 采用原码表示 11011。

采用原码作乘除运算比较方便,可取绝对值直接运算,而符号单独处理。但对于应用最多的加减运算,原码表示不太方便,像  $(-2)+3$  表面作加法操作,而实际需作  $(3-2)$  减法操作。

### (二) 数的补码表示

补码表示能让符号位一同参与运算,将减法化成加法。

正数的补码就是本身,形式与原码相同。负数的补码符号与原码相同,其余各位取反,然后末位加 1。

+1011 采用补码表示为 01011,-1011 采用补码表示为 10101。

$[x-y]_n = [x]_n + [-y]_n$ ,符号位同时参加运算。

如何从补码求原码或真值,可按求补码同样的方法进行。

$n$  位补码表示的数的范围为  $[-2^{n-1}, 2^{n-1}-1]$ 。

为扩大数的表示范围,可用两个字来表示一个数,这种数称为双字长数或双精度数。

在计算机中,负数一般用补码表示。

### (三) 十进制数的二进制码

十进制数的二进制编码称为 BCD 码。它的引入是为解决日常习惯的十进制与机器内的二进制之间的矛盾,方便进行十进制与二进制之间的转换,最常用的是 8421 码。它对每一位十进制数码用 4 位二进制编码表示。

十进制数 1329 用 BCD 码表示为:0001 0011 0010 1001

BCD 码在 0 至 9 的范围与纯二进制无区别,不同之处仅在于逢十进位,而 4 位纯二进制数逢十六进位,因此对大于等于 10 的数需作加 6 修正。

BCD 码有压缩 BCD 码和非压缩 BCD 码二种形式,压缩 BCD 码的每个十进制数字按四个二进制位一组存放,而非压缩 BCD 码的每个十进制数字占每个字节的低 4 位,高 4 位内容是什么不重要。

### (四) 字符编码

ASCII 码是美国信息交换标准代码,是最主要的字符编码方式。对字符进行编码是非数值处理,实现输入输出的基本手段。

字符包括字母、数字、专用字符及一些非打印字符。一个字符用 8 位来表示,其中低 7 位为字符的 ASCII 码值,最高位是校验位,7 位编码的 ASCII 码能表示 128 种字符。参见 ASCII 码表。

数字的 ASCII 码用数字加十六进制数 30 可得到。当按某个键,就产生相应的 ASCII 码并送至计算机。

### (五)几点说明

在某些情况下,要处理的数据全是正数,此时保留符号位就无意义,可将最高有效位也作为数值位处理,形成无符号整数。8 位无符号整数范围是 0 至 255,16 位无符号整数范围是 0 至 65535。计算机中常用无符号整数表示地址。

数据在计算机内部采用何种方式,依赖于程序的执行情况,可用二进制、二—十进制、字符方式。无符号整数 1996 以压缩 BCD 码、未压缩 BCD 码、二进制方式表示,分别需 16、32、11 位。

数据的存储是以数据长度为单位,按字节顺序存放,用低端低地址方式来存储大于一个字节的数,低位字节放低地址,高位字节放高地址。

数据要扩展位数,一般是对当前最高位(符号位)进行扩展。当前最高位为 0,扩展位补 0,当前最高位为 1,扩展位补 1。

## 第三节 计算机系统结构

计算机系统结构是指计算机主要部件的特性布局及连接,是程序设计者必须了解的计算机资源及其资源部件,如地址空间、寄存器组、寻址方式、指令系统等。典型的计算机结构包括 CPU、存储器、输入/输出系统三个主要部分,通过系统总线连接。CPU 是整个系统的核心,指令的执行、机器的工作都由其控制完成。各部分可动态构成各种规模系统。

现在广泛装备的是 PC 系列微机,其 CPU 主要为 Intel 公司所提供的 80486(80X86)系列微处理器,本教材介绍的汇编语言程序设计即建立在其基础上。

### 一、80X86 微处理器

80X86 是以 Intel 公司 1978 年推出的 8086 微处理器为基础,作纵横两个方向扩展发展起来的微处理器系列,其 CPU 为 8086/8088、80286、80386、80486,至目前最新的奔腾与高能奔腾中的一种,配以适当数量的支持芯片,构成功能强大的微处理器系统。

#### (一)8086/8088

8086CPU 是 16 位微处理器,采用 16 位数据总线。8088CPU 是准 16 位的微处理器,采用 8 位数据总线,而使用 16 位内部结构。8086 具有 6 字节指令流队列,8088 则是 4 个字节。两者指令系统相同,汇编语言一致,除了运行速度,其他方面无区别。用以装备早期及低档微机。

##### 1. 8086CPU 结构

8086CPU 由指令执行部件 EU 与总线接口部件 BIU 两部分组成,图 1-1 为其结构示意图。

EU 部件(Execution Unit)控制和执行指令,主要由算术逻辑运算部件 ALU、EU 控制部件、8 个 16 位寄存器加标志寄存器组成。

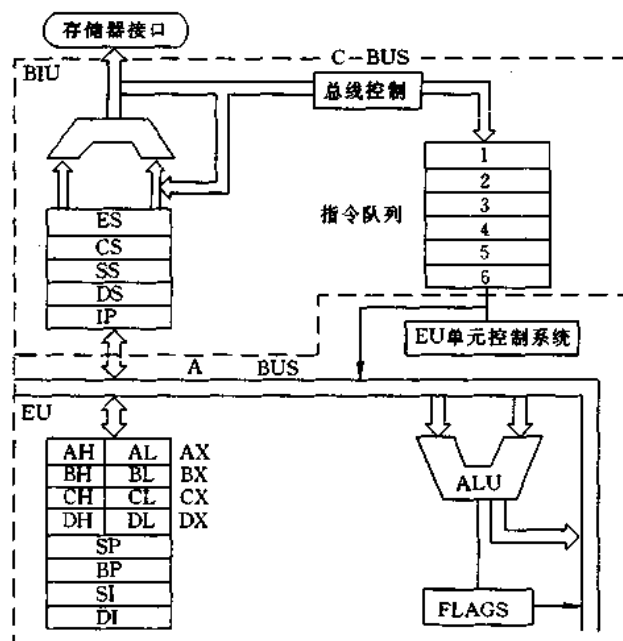


图 1-1 8086 CPU 结构示意图

BIU 部件 (Bus Interface Unit) 负责从存储器预取指令和数据, 以及所有 EU 需要的总线操作, 实现 CPU 与存储器和外设之间信息传递。主要由指令队列、指令指针寄存器、寄存器、地址加法器组成。

EU 和 BIU 能独立运行, 在一条指令的执行过程中, 就可取下一条指令送指令队列, 实现并行操作。

## 2. 内部寄存器

不包括数据暂存器, 内部寄存器可分为四组。

### (1) 数据寄存器

数据寄存器包括 AX、BX、CX、DX 四个 16 位通用寄存器。主要用于存放操作数和运算结果。其高 8 位与底 8 位也可以分开作为 8 位的寄存器使用, 高 8 位用后缀 H 表示, 低 8 位用后缀 L 表示。8 位寄存器有 AH、AL、BH、BL、CH、CL、DH、DL 八个。

AX、BX、CX、DX 还有各自的专用用途。

AX (Accumulator) 是算术运算中主要寄存器, 在乘除运算中作累加器用, 指定存放被乘数与被除数。另外所有 I/O 指令使用这一寄存器与外部设备传送信息。

BX (Base) 在计算存储器地址时, 经常用作基址寄存器。

CX (Count) 在循环和串处理指令中用作隐含的计数器, CL 还可作为移位计数器。

DX (Data) 在双字长运算时与 AX 组合, 存放高位字。也常用来存放 I/O 端口地址。

### (2) 指针及变址寄存器

指针及变址寄存器包括 SP、BP、SI、DI 四个 16 位寄存器, 可像数据寄存器一样, 作为存放操作数的通用寄存器, 更经常的用途是存放地址, 提供寻址时的偏移地址, 用于形成指令要访问的存储器地址。

SP (Stack Pointer) 是堆栈指针寄存器, 用来指示栈顶的偏移地址。BP (Base Pointer) 是基址指针寄存器, 可作为堆栈的基地址。SP 或 BP 与段寄存器 SS 结合, 可确定堆栈中某一存储

单元地址。

SI(Source Index)和 DI(Destination Index)是源变址寄存器和目的变址寄存器,一般与段寄存器 DS 联用,用来确定数据段中某一存储单元地址。

### (3)段寄存器

段寄存器包括 CS、DS、SS、ES 四个 16 位寄存器,分别用来标识代码段、数据段、堆栈段和附加段的段地址,用以扩大可访问的存储空间,实现 CPU 对内存不同段的读写。参见稍后的存储分配方式。

### (4)控制寄存器

控制寄存器包括指令指针寄存器 IP 和标志寄存器 FLAGS 两个 16 位的寄存器。

IP 又称程序计数器,它存放当前要执行指令的地址,由于 8086 采用分段存储程序和数据,IP 实际上存放的是当前代码段的偏移量,它随程序的执行而变化,硬件上保证总是自动加 1,计算机用 IP 寄存器来控制指令序列的执行流程。

FLAGS 又称程序状态字寄存器,它的内容标志着程序执行时 CPU 的状态,8086 只使用 16 位中的 9 位,分布如下:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OF	DF	IF	TF	SF	ZF		AF		PF		CF

9 个标志位分成状态标志和控制标志两组。状态标志位寄存着指令执行结果的状态信息,往往作为后续条件转移指令的控制条件,亦称为条件码标志位,包括以下 6 位:

OF,溢出标志位。运算结果超出系统所能表示的数的范围称为溢出,此时 OF 位置 1,否则置 0。对于无符号数的运算必须根据进位标志位 CF 决定是否溢出。

SF,符号标志位。记录运算结果的符号,结果为负时置 1,否则置 0。与运算结果最高位始终保持一致。

ZF,零标志位。运算结果为 0 时,置 1,否则置 0。

AF,辅助进位标志。运算第 3 位(半字节)产生的进位或借位时置 1,否则置 0。该标志仅用于控制十进制调整指令。

PF,奇偶标志位。运算结果中 1 的个数为偶数置 1,否则置 0。常用来检查数据在传送过程中是否发生错误。

CF,进位标志位。运算时最高位产生进位或借位时置 1,否则置 0。该标志主要用于多字节数的加减运算。

控制标志位对某些 CPU 的操作进行干预,包括以下 3 位:

DF,方向标志位。标志字符串指令处理的方向,DF 为 1 时,串处理从高地址向低地址递减进行,DF 为 0 时,串处理从低地址向高地址递增进行。

IF,中断标志位。IF 为 1 允许中断,IF 为 0 屏蔽中断。

TF,自陷标志位。TF 为 1 进入单步执行方式,以便调试程序,TF 为 0 是连续执行方式。

## (二)80286

80286CPU 在 1984 年推出,采用 16 位数据总线,支持实模式与保护模式操作,在保护模式下可访问 16M RAM,用以装备中低档微机。

## 1. 80286CPU 结构

80286CPU 由指令执行部件 EU、指令译码部件 IU、总线部件 BU、地址部件 AU 四个部分组成。各部分能平行工作,从而减少了彼此等待时间,提高了系统的效率。图 1-2 为其结构示意图。

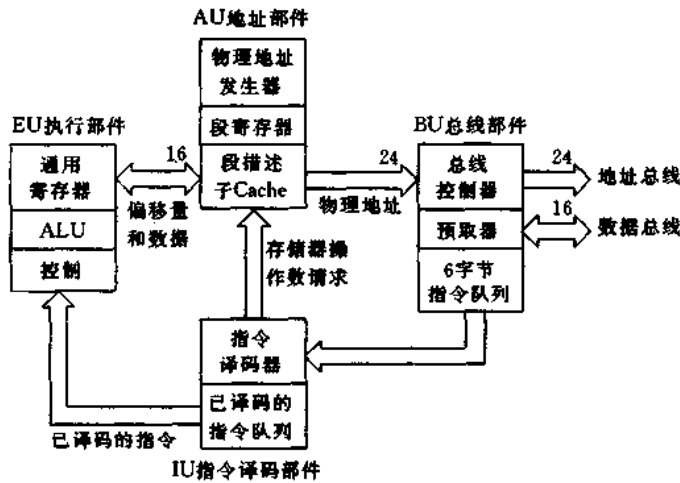


图 1-2 80286CPU 结构示意图

EU 部件与 8086 类似,但它不对指令译码。指令译码由指令译码部件 IU 完成,IU 部件可存放 3 条已译码的指令,并负责所有总线之间的通信和数据传输。AU 部件在实模式下与 8086 相应部分功能一样,在保护模式下,它支持虚拟存储方式及存储保护。

## 2. 内部寄存器

80286 的寄存器与 8086 的寄存器只有以下几点不同:

(1)标志寄存器增加了两个标志,占用三个标志位,用于保护模式。

IOPL、IO,特权标志,占用第 12 及 13 位,表示该任务使用的 IO 操作所处特权级。特权共 4 级,00 为最高级,11 为最低级。

NT,任务嵌套标志,占用第 14 位。NT 为 1,表示当前执行任务嵌套在另一任务中,实模式下 NT 为 0。

(2)增加了一个 16 位机器状态字寄存器 MSW。

MSW 目前使用低 4 位。其中 PE 是操作模式位,置 1 时为保护模式,否则为实模式。MP 是协处理器标志位,置 1 表示协处理器存在,否则不存在。EM 是仿真协处理器标志位,置 1 要求用软件仿真协处理器,否则不要求。TS 是任务转换标志位。

另外还增加有 4 个寄存器用以支持保护虚存机构。

## (三)80386

80386CPU 在 1986 年推出,有 DX 与 SX 两种型号,在内部都采用 32 位总线,但外部 DX 的数据与地址总线都是 32 位,SX 的外部地址总线是 24 位,数据总线 16 位。支持实模式与保护模式操作,在保护方式下可访问 4G RAM。用以装备中档微机。

## 1. 80386CPU 结构

80386CPU 由指令执行部件、指令预取部件、指令译码部件、总线部件、分段部件、分页部件六个部分组成。各部件间均能并行操作,CPU 同一时间可处理多条指令的不同阶段,各部件



都能和总线部件直接通信,CPU 的能力大为提高,图 1-3 为其结构示意图。

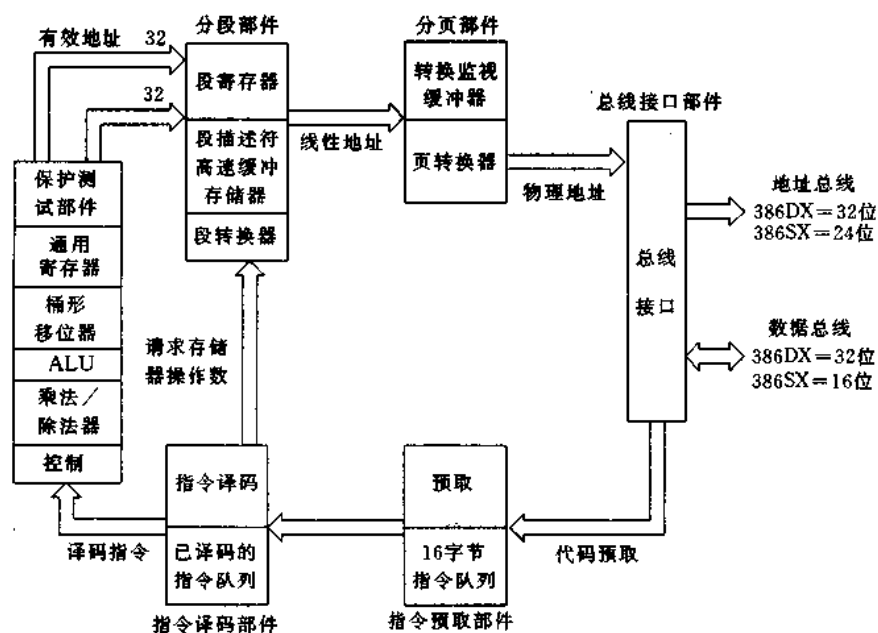


图 1-3 80386CPU 结构示意图

总线部件与 80286 的总线部件功能类似,不同的是预取指令功能从总线部件分离出来成为一个独立的处理部件。指令预取部件在 16 个字节的预取指令队列有 2 个字节空闲时就向总线请求取指令,但优先级最低。指令译码部件与 80286 功能类似。指令执行部件负责执行指令,包含有 1 个 64 位的桶形移位器,能在 1 个时钟周期实现多位的移位。分段部件与 80286CPU 的地址部件功能类似,但加速了地址的计算速度,并且把线性地址传送给分页部件,而不是送给总线。分页部件的功能是将线性地址转换成物理地址,转换有两种情况,如果无分页特征,则该线性地址就是物理地址,如果有分页特征,则将线性地址按页存储方式转换为物理地址,页长为 4KB,最后将物理地址送总线。

## 2. 内部寄存器

80386 采用 32 位寄存器,包括 8 个通用寄存器 EAX、EBX、ECX、EDX、ESP、EBP、ESI、EDI,低 16 位及其低 8 位与高 8 位可同 8086 的寄存器完全相同使用(包括名称),但高 16 位不能单独使用。32 位寄存器的使用方法也类似,但增强通用性(包括对应 16 位寄存器),除 SP 寄存器外,均可作基址寄存器或变址寄存器使用。

80386 的指令指针寄存器 EIP 可存放 16 位或 32 位地址。标志寄存器 EFLAGS 也是 32 位的寄存器,包含 80286 标志寄存器各位的定义,并增加了 2 个标志位,重新开始标志 RF 占用第 16 位,用于控制调试异常中断。虚拟 86 模式标志占用第 17 位,置 0 为保护模式下运行,置 1 为虚拟 86 模式下操作。

另外增加 FS、GS 两个数据段寄存器,共 6 个 16 位段寄存器,可同时用于 6 个分段。还有机器状态字寄存器、保护模式管理寄存器及调试寄存器等。

## 二、80486 微处理器功能结构

80486 是 Intel 公司在 1989 年推出的高性能微处理器,采用完整的 32 位体系结构,CPU