汇编语言程序

设计

高等专科学校教材

汇编语言程序设计

冯子纲　罗万钧　朱建华

西安电子科技大学出版社

内 容 简 介

本书系统地介绍了汇编语言程序设计的基本理论和方法，并附有完整的上机实习辅导资料和习题，是一本理论和实践紧密结合的计算机专业大专教材。

全书共10章。第一、二、三章主要介绍计算机基础知识、Z80的指令系统及其汇编语言；第四、五、六、七章系统介绍了分支程序、循环程序、子程序和宏程序的设计方法；第九章介绍了Intel 8086/8088指令系统及其汇编语言；第八章和第十章介绍了输入输出程序设计，Z80、8088汇编语言源程序的开发过程和常用程序的设计实例。强调汇编程序的应用开发是本书的特点之一。

本书既可作为大专计算机软件及应用专业的教材，也可作为从事计算机专业的科技人员的参考书。

高等专科学校教材

汇编语言程序设计

冯子纲　罗万钧　朱建华

责任编辑　徐德源

西安电子科技大学出版社出版

西安电子科技大学印刷厂印刷

陕西省新华书店发行　各地新华书店经售

开本 787×1092　1/16　印张 22 12/16　字数 556 千字

1988 年 6 月第 1 版　1988 年 6 月第 1 次印刷　印数 1—7 000

ISBN 7-5606-0036-0/TP·0012　　定价：3.80元

出 版 说 明

根据国务院关于高等学校教材工作分工的规定，我部承担了全国高等学校、中等专业学校工科电子类专业教材的编审、出版的组织工作。由于各有关院校及参与编审工作的广大教师共同努力，有关出版社的紧密配合，从1978年至1985年，已编审、出版了两轮教材，正在陆续供给高等学校和中等专业学校教学使用。

为了使工科电子类专业教材能更好地适应“三个面向”的需要，贯彻“努力提高教材质量，逐步实现教材多样化，增加不同品种、不同层次、不同学术观点、不同风格、不同改革试验的教材”的精神，我部所属的七个高等学校教材编审委员会和两个中等专业学校教材编审委员会，在总结前两轮教材工作的基础上，结合教育形势的发展和教学改革的需要，制订了1986～1990年的“七五”（第三轮）教材编审出版规划。列入规划的教材、实验教材、教学参考书等近400种选题。这批教材的评选推荐和编写工作由各编委会直接组织进行。

这批教材的书稿，是从通过教学实践、师生反映较好的讲议中经院校推荐，由编审委员会（小组）评选择优产生出来的。广大编审者、各编审委员会和有关出版社为保证教材的出版和提高教材的质量，作出了不懈的努力。

限于水平和经验，这批教材的编审、出版工作还会有缺点和不足之处，希望使用教材的单位、广大教师和同学积极提出批评建议，共同为不断提高工科电子类专业教材的质量而努力。

电子工业部教材办公室

前　言

本教材系按电子工业部制定的工科电子类专业教材1986～1990年编审出版规划，由大专计算机专业教材编审委员会专业基础编审小组组织征稿、评选、推荐出版的。

本教材由西北电讯工程学院苏州分部(苏州电子工业职工大学)冯子纲同志主编，上海第二工业大学江庚和教授主审。

本课程的参考学时数为80学时，主要包括两部分内容：Z80指令系统和Intel8086/8088指令系统，前者占主要篇幅，而后者只占部分章节。本教材选用这两种指令系统是在充分考虑了当前国内的实际情况和大专教材的特点后确定的。目前，以Z80CPU组成的单板机及其系统机在我国占多数。由于Z80CPU及其配套芯片价格低廉，并能根据实际需要灵活地组成较小的系统，故被广泛地用于工业加工、过程控制和各类智能仪器中。因此掌握Z80汇编语言程序设计的基本方法和技能，有着非常大的实用价值。Z80指令系统功能齐全，使用灵活方便、规模适中、易于掌握，在8位机中最具代表性。实践证明，掌握了Z80汇编语言程序设计方法，可以毫不费力地在较短时间内掌握其他8位机的汇编语言程序设计方法，同时也为学习和掌握高档微型机(各种16位机及32位机)的汇编语言程序设计打下了坚实的基础。近年来，以IBM-PC及其兼容机(如国产长城0520系列机)为代表的16位微型机以很快的速度在我国推广普及，不久将要成为我国正在使用的微型机的主要品种之一。与8位机相比，无论是在存贮容量，寻址能力、运算速度、指令功能上，还是在多处理机连接方式上，它都具有明显的优势。因此，介绍IBM-PC机的CPU，即8088(与8086兼容)的指令系统及其汇编语言也是非常必要的。

本教材充分注意到大专计算机专业的特点，在强调基础理论的同时，把加强实践性环节，学以致用提到足够的高度予以重视，选定Z80及8086/8088这两种最具代表性又最实用的指令系统，正是从学以致用的角度考虑的。教材中编入了完整的上机实习辅导资料及习题集，利用它们可方便地进行汇编语言源程序的编写、汇编、连接、调试和运行，在操作系统支持下可以实现软件工作者所要实现的工作。第一、二章是后续各章的基础，第三章至第七章是从程序的基本结构上全面介绍指令具体用法的。只有掌握一定数量的常用程序，才能熟练地掌握各种基本程序结构，也才能掌握汇编语言程序设计的基本方法和技能，为此在各章节中分批介绍了各类常用的算法，它们可以作为程序设计的借鉴。本书不仅可以作为大专教材，亦可作为专业工作者的参考书。

冯子纲编写了本教材的第一至第六章及第九章，苏州电子工业职工大学罗万钧同志编写第七、第八、第十章，南京化工动力专科学校朱建华编写习题集、部分上机资料并提供第五、第六章的初稿，最后由罗万钧统编全稿。参加审阅工作的还有上海工业大学的应振时副教授、林显忠讲师。他们对本书提出了许多宝贵意见，在此向他们表示诚挚的感谢。由于编者水平有限，书中难免存在一些缺点和错误，殷切希望广大读者批评指正。

冯子纲　罗万钧　朱建华

1986年12月

目　录

第一章 计算机基础

自从1946年第1台电子计算机问世以来，计算机已经历了电子管计算机、晶体管计算机、中小规模集成电路计算机3代的更新而发展到当前的大规模集成电路或超大规模集成电路的第4代计算机。微型计算机在70年代后期有了飞速发展。由于微型机价格低廉，性能优越，大有取代中、小型机的趋势。

§1-1 计算机的基本组成

到目前为止，从组成看，现已推出的计算机都属于冯·诺依曼型的计算机，均由控制器、运算器、存贮器(内部)，输入设备和输出设备5大部分组成。如图1-1所示。

输入设备：人与计算机交往的入口。常用的输入设备有键盘、纸带输入机等。

输出设备：计算机与人交往的输出窗口。它把运算结果或各种信息以数字、字符、图形等形式表示出来。常用的输出设备有打印机、纸带穿孔机、数码管、阴极射线管(CRT)显示器和绘图仪等。

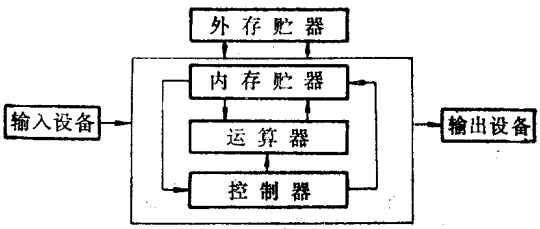

图1-1 电子计算机组成示意图

存贮器：寄存数据、程序的部件。计算机的存贮器分内存贮器和外存贮器两大类别，用作内存贮器的有磁芯存贮器和半导体存贮器，用作外存贮器的有磁带机、磁鼓和磁盘等。有时把外存贮器与输入、输出设备统称为外部设备或外设，有时却把外设与输入、输出设备(I/O设备)看成是一回事。

存贮器使计算机具有记忆信息的能力。这些信息可以是一系列的指令代码的集合(称为计算机程序或机器代码程序)，也可以是原始数据或运算结果(包括中间计算结果)。存贮信息的物理装置叫存贮单元，这些存贮单元的编号叫地址。计算机对某一存贮单元存(写)、取(读)信息是按地址进行的。

运算器：计算机对各种信息进行算术运算和逻辑运算的主要部件。

控制器：计算机的指挥部，它控制整个计算机自动、协调一致地工作。

通常把运算器和控制器合在一起称为中央处理器(CPU)。

§1-2 电子计算机中的数制和码制

一、电子计算机中采用的数制

数制也称为进位计数制。日常生活中，人们习惯采用10进制进行运算，而计算机内的所有信息则以2进制的代码表示。

(一) 10进制数的特点

(1) 它有 10 个不同的符号(或称为元素、系数)，即 0,1,2,3,4,5,6,7,8,9。

(2) 它是逢 10 进位的，因为一位 10 进制数只能为 10 个不同的符号(状态)，即 0～9 中的一个，因此，对于大于"10"的数，就要用多位 10 进制数来表示。例如，一个 10 进制数为：1234.56，其中的"4"代表个位的 4，即它自身。而其中的"3"，由于它处在十位的位置上，因而它所代表的数已不是"3"本身，而是 30 或 3×10^{1}。同理，其中的"5"，代表 5×10^{-1}。我们可以把它表示成如下形式：

$$1234.56=1\times10^{3}+2\times10^{2}+3\times10^{1}+4\times10^{0}+5\times10^{-1}+6\times10^{-2}$$

一般地说，任意一个 10 进制数 N，都可以表示为

$$N=\pm[k_n\times10^{n}+k_{n-1}\times10^{n-1}+\cdots+k_1\times10^{1}+k_0\times10^{0}+k_{-1}\times10^{-1}+k_{-2}\times10^{-2}+\cdots+k_{-m}\times10^{-m}]$$

$$=\pm\sum_{i=-m}^{n}[k_i\times10^{i}]$$

其中 k_i 为 0,1,2,3,4,5,6,7,8,9。

(二) 数的一般表示法

对不同数制，N 可表示为

$$N=\pm\sum_{i=-m}^{n}[k_i\times R^{i}]$$

R 称为进位制的基数，对于 10 进制数，$R=10$，在 2 进制中，$R=2$；8 进制中，$R=8$；16 进制中，$R=16$ 等等。

R^{i} 称为第 i 位的位权，不同进位制中的位权是不一样的，同一数制中不同位的位权也不相同。

表 1-1　常用进位计数制的表示方法

10 进制数	2 进制数	8 进制数	16 进制数
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A 或 $\overline{0}$
11	1011	13	B 或 $\overline{1}$
12	1100	14	C 或 $\overline{2}$
13	1101	15	D 或 $\overline{3}$
14	1110	16	E 或 $\overline{4}$
15	1111	17	F 或 $\overline{5}$
16	10000	20	10

k_i 称为第 i 位的系数，在 R 为基数时，$k_i=0,1,\cdots,R-1$。

例如：$R=2$，$k_i=0,1$

$R=8$，$k_i=0,1\cdots,7$

$R=10$，$k_i=0,1,\cdots,9$

$R=16$，$k_i=0,1,\cdots,15$

常用的几种进位计数制的表示方法见表 1-1。

（三）各种数制间的转换

1．10 进制数与其他计数制之间的转换

在 10 进制数与 2 进制数之间的转换中，整数转换与小数转换的方法是不一样的。下面分别叙述之。

10 进制整数转换成 2 进制整数采用除 2 取余法。

例 1 将 10 进制数 26 转换成 2 进制数。

解：

2	26	余 数
2	13	$0=k_0$
2	6	$1=k_1$
2	3	$0=k_2$
2	1	$1=k_3$
	0	$1=k_4$

于是$(26)_{10}=k_4k_3k_2k_1k_0=(11010)_2$，其中$(X)_n$表示 X 是 n 进制数。

除 2 取余法是将$(26)_{10}$这个数不断地除以 2，除尽无余数的 $k_i=0$；除不尽有余数的 $k_i=1$。

现在我们再验算一下：

$$(11010)_2=1\times2^4+1\times2^3+0\times2^2+1\times2^1+0\times2^0$$
$$=16+8+0+2+0=(26)_{10}$$

把10 进制纯小数转换成 2 进制的纯小数用乘 2 取整法。

例 2 将 10 进制数 0.625 转换成 2 进制数。

解：

$$\begin{array}{r} 0.625 \\ \times)\quad 2 \\ \hline 1.250 \end{array}$$ 得 $k_{-1}=1$

$$\begin{array}{r} 0.25 \\ \times)\quad 2 \\ \hline 0.50 \end{array}$$ 得 $k_{-2}=0$

$$\begin{array}{r} 0.5 \\ \times)\quad 2 \\ \hline 1.0 \end{array}$$ 得 $k_{-3}=1$

故 $(0.625)_{10}=(0.k_{-1}k_{-2}k_{-3})_2=(0.101)_2$

将小数反复乘 2，若所得新数的整数部分为 1，则相应位为 1；若整数部分为 0，则相应位为 0，从高位向低位逐次进行，直到满足精度要求为止。

验算：$(0.101)_2 = 1\times2^{-1}+0\times2^{-2}+1\times2^{-3} = (0.625)_{10}$

当10进制混合小数转换成2进制数时，可采用整数、小数部分分别转换的办法。它们的逆运算，即2进制数转换成10进制数，前面已作过介绍，用通用表达式展开求之。

小结：10进制整数转换成2进制整数用“除2取余法”；10进制小数转换成2进制小数用“乘2取整法”；2进制数转换成10进制数用通用表达式展开法。

不难验证，将10进制整数转换成8进制整数用“除8取余法”；10进制纯小数转换成8进制纯小数用“乘8取整法”；而8进制数转换成10进制数用通用表达式展开法，例如：

$$(1503.54)_8 = 1\times8^3+5\times8^2+0\times8^1+3\times8^0+5\times8^{-1}+4\times8^{-2} = (835.6875)_{10}$$

同样，10进制整数转换成16进制整数用“除16取余法”，10进制小数转换成16进制小数用“乘16取整法”，而16进制数转换成10进制数用通用表达式展开法。

2．2进制与8进制之间的转换

这种转换以小数点为界，整数部分自右向左每3位分1组，不够则前加零；小数部分自左向右每3位分1组，不够则后加零，然后按2进制与8进制的规律直接阅读。例如：

$$(1101001.0100111)_2 = (001\ 101\ 001.010\ 011\ 100)_2 = (151.234)_8$$

8进制数到2进制数的转换是上述转换的逆过程。例如

$$(653.503)_8 = (110101011.101000011)_2$$

按8进制与2进制的规律阅读。

3．2进制与16进制之间的转换

因为4位2进制数表示1位16进制数，所以在16进制与2进制之间也存在类似8进制与2进制之间的简单而又直接的转换规则，即以小数点为界，整数部分自右向左每4位分1组，不够则前加零；小数部分自左向右每4位分1组，不够则后加零，然后按2进制与16进制的规律阅读。例如：

$$(16D.44)_{16} = (000101101101.01000100)_2$$
$$= (101101101.0100010)_2$$

同样，按16进制与2进制的规律阅读。

（四）10进制数的2进制编码——BCD码

10进制数可用2进制的编码来表示，这种编码叫做BCD码。一位BCD码用4位2进制编码来表示。编码的方法很多，最常用的编码是8421BCD码。即最高位的权为8，依次逐位降低，其权分别为4、2及1。BCD码的编码如表1-2所示。显然，BCD码比2进制数直观。

Z80指令系统中有一条10进制调整指令DAA，它使计算机可以进行用BCD码表示的10进制数的加法和减法运算。

表1-2 8421BCD码编码表

10进制数	8421BCD码
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
10	0 0 0 1 0 0 0 0
11	0 0 0 1 0 0 0 1
12	0 0 0 1 0 0 1 0
⋮	⋮ ⋮
19	0 0 0 1 1 0 0 1
⋮	⋮ ⋮

二、计算机中的码制

（一）机器数和真值

一个数若不考虑它的符号，即为无符号数。具

体的数显然有正有负，用数学符号“+”或“-”表示，如 $N=+91$ 或 $N=-91$。

通常把带有正负号的数称为真值。但是，计算机中所有的数都用 0 或 1 进行编码，机器并不认识正负号，只有将正负号也变为 0 或 1，机器才能认识。例如 91 的 2 进制数表示为 1011011，若前面加一位符号位，以 0 代表正，1 代表负，则 +91 可用 01011011 来表示，而 -91 可用 11011011 来表示。我们把带有符号位的 2 进制数称为机器数。机器数和它的真值之间存在着一一对应的关系。

为了运算方便，机器数有 3 种表示法，即原码、反码和补码。

（二）原码表示法

定义： $[X]_{原}=\begin{cases} X & 0\leqslant X<2^{n-1} \\ 2^{n-1}-X & -2^{n-1}<X\leqslant 0 \end{cases}$

其中 n 为 2 进制数的位数。

从上面的定义可以看到：

当 $X>0$，设 $X=+1001010$ 则 $[X]_{原}=01001010$，即正数的原码就是它自己；

当 $X<0$，设 $X=-1001010$ 则 $[X]_{原}=11001010$，即负数的原码符号位取 1，数值部分不动。

上面介绍的是整数的原码表示法，小数的原码表示法将另作定义。

小结：

(1) 正数原码的符号位用 0 表示，负数原码的符号位用 1 表示。

(2) 零有两种情况，即

$$[+0]_{原}=0\ \underbrace{0\ 0\cdots 0}_{(n-1)个0}$$

$$[-0]_{原}=1\ \underbrace{0\ 0\cdots 0}_{(n-1)个0}$$

机器中均作零处理。一般采用正零表示法。

(3) 当 $n=8$，即对 8 位机讲，数的原码表示范围是 $-127\sim+127$。

（三）补码表示法

我们知道两个正数相加，其和为正数，两个负数相加，其和为负数，而当一个正数与一个负数相加时，必须做减法，和的符号位与绝对值较大的那个数的符号相同，因此必须判断绝对值的大小，这样就必然使控制线路变得更加复杂。采用补码表示法时，正负数的加减运算被转化为单纯的补码相加运算，从而使问题变得简单。

日常生活中，补码的例子是很多的，时钟的校时就是其中的一例。设标准时间是 6 点正，而时针指向 10 点正，要将它拨到 6 点钟，可有两种拨法：

(1) $10-4=6$　采用倒拨的办法实现；

(2) $10+8=6$　采用顺拨的办法实现。

这儿倒拨与顺拨的效果是相同的。

上述结果，用数学语言来描述，就称 -4 与 +8 对模 12 讲是互补的，或者说 -4 与 +8 对模 12 讲是同余的。用数学表达式可写成：

$$-4=8 \pmod{12}$$

$$6 = 18 \pmod{12}$$

此处的模是指一个计数系统的量程或此计数系统所能表示的最大的数。显然，时钟的模为 12，所以 18 与 6 是同余的，或者说是互补的。在模 12 系统中 18 只能用 6 表示，因此，对正数讲，它的补码就是它自身。但对负数讲，它的同余数可能比模 12 小，例如 -4 的同余数 8 就比 12 小。因 $12+(-4)=8$，所以，负数的补码值为模加负数本身。至此，我们可以对模 2^n 系统的 2 进制补码定义如下：

定义： $[X]_{补}=\begin{cases} X & 0\leqslant X<2^{n-1} \\ 2^n+X & -2^{n-1}\leqslant X<0 \end{cases}$

其中 n 为 2 进制数的位数。当 $n=8$，补码表示的数的范围是 $-128\sim+127$。

例 3 已知 $X_1=(01000000)_2$，$X_2=(-1010)_2$，求 $[X_1]_{补}$、$[X_2]_{补}$。

解： $[X_1]_{补}=X_1=(01000000)_2$

$[X_2]_{补}=2^8+X_2=2^7+2^7-(1010)_2=(11110110)_2$

从例 3 可以看出，对负数求补码时，需要作一次减法，这是很不方便的。现在给出一个求补码的简便方法，其证明可参考有关文献。

负数的补码，等于其原码除符号位外按位求反，然后再加 1。

例如，在上例中 $[X_2]_{原}=(10001010)_2$，按求补码的简便方法，有

$$[X_2]_{补}=11110101+1=(11110110)_2$$

这与按定义求得的结果是一致的。

下面给出 3 个与补码有关的公式，此处不加证明，有兴趣的读者可查阅有关资料。

(1) $[X-Y]_{补}=[X]_{补}+[-Y]_{补}$

(2) $[[X_{补}]]_{补}=X$

(3) $[-Y]_{补}=[[Y_{补}]]_{求补}$

此处，求补操作是指将原码连同符号位一起按位求反，然后再加 1。

例 4 求 $X_2=(-1010)_2$ 的补码。

解： $[X_2]_{补}=[[1010]_{补}]_{求补}=[00001010]_{求补}=11110101+1=(11110110)_2$

（四）反码表示法

定义： $[X]_{反}=\begin{cases} X & 0\leqslant X<2^{n-1} \\ (2^n-1)+X & -2^{n-1}<X\leqslant 0 \end{cases}$

从定义可知：

(1) 正数的反码就是它自己，负数的反码为符号位取 1，数值部分取反。

(2) 零有两种情况。

$$[+0]_{反}=0\ \underbrace{0\cdots0}_{n\text{个}0}$$

$$[-0]_{反}=1\ \underbrace{1\cdots1}_{n\text{个}1}$$

(3) 当 $n=8$，即对 8 位机时，数的表示范围 $-127\sim+127$。

(4) 当一个带符号的数用反码表示时，最高位为符号位。当符号位为0(即正数)时，后面的7位为数值部分，但当符号位为1(即负数)时，一定要注意后面的7位表示的不是此负数的数值，只有把它们按位取反，才表示它的2进制值。例如，若10010100是一个负数的反码，则它不等于$(-20)_{10}$，而等于$(-107)_{10}$。

三、字符数据及其编码

包括字母在内的各种字符，如数码0～9，字母A～Z，符号+、－、×、/、·等也必须按特定的规则用2进制编码才能在机器中表示。编码可有各种方式，普遍采用的ASCII码(美国标准信息交换码)用7位2进制编码，可表示128个字符，包括数码0～9，字母A～Z。数码0～9是用30H～39H表示的，字母A～Z是用41H～5AH表示的，详见附录Ⅰ。

四、代码书写形式

(一) 计算机字长

如前所述，计算机内所有的信息都是以2进制的代码的形式来表示的，这个2进制代码的位数称为计算机的字长。计算机位数愈多，它能代表的数值就愈大，能表示的数值的有效位数也愈多，计算机的精度也愈高。但是，字长愈长，计算机电路也就愈复杂。大型机字长一般都是32位或大于32位，中型机字长一般是32位或16位，小型机字长几乎都是16位，而微型计算机的字长则有1位、4位或8位，现在已有16位及32位微型机投入使用。就我国目前情况，大量使用的微型机字长是8位的，16位字长的微型机正在迅速地普及，32位字长的微型机也已进入计算机市场。

(二) 代码书写形式

若计算机字长为8位，则一个代码，就要用8个0,1的编码来组成，这样不仅书写困难，不便阅读，而且容易出错。因而通常用8进制或16进制数来书写。2进制数与8进制、16进制数之间的转换，前面已经介绍了。为了区别各种数制表示的代码，在2进制数后面加一个字母B(Binary)，例如01011101B；在8进制数的后面加一个字母O(Octal)，例如036O；在16进制数的后面加一个字母H(Hexadecimal)，例如7EH；在10进制数的后面加一个字母D(Decimal)，字母D也可省略不写，但在任何情况下，8进制数与16进制数后一定要加上字母O及H。表1-3是各种进位制数的书写形式与2进制数的对应关系。

表1-3 各种进位制书写形式对照表

各种进位制形式	2进制形式
01011101B	0 1 0 1 1 1 0 1
036O	0 0 0 1 1 1 1 0
7EH	0 1 1 1 1 1 1 0
68D	0 1 1 0 1 0 0 0
36	0 0 1 1 0 1 1 0

(三) 信息单位——位、字节、字、双字

(一) bit(位)：将一位2进制位叫1bit，它是最小的信息单位。

2. Byte(字节)：将 8 位 2 进制位叫一个字节，1 Byte = 8 bit。

3. Word(字)：将 16 位 2 进制位或两个字节称为一个字，即 1Word = 2 Byte = 16 bit。

4. Doubleword(双字)：将两个字称做双字，并简记为 DW。显然，1 DW = 2W = 4B = 32 bit。

位、字节、字、双字是经常碰到的信息单位。

§1-3 机器语言·汇编语言·高级语言

一、指令和程序

冯·诺依曼型计算机的特点是程序的存贮和自动执行。在计算机执行某一任务之前，首先要为这一任务编写一个程序。通俗地讲，程序就是计算机执行任务的步骤或操作过程，它又是由一系列能完成某一操作的指令组成的，因此，程序是指令的有序集合。在编写好程序以后，还要把它事先存到内存贮器的程序区(或称指令代码区)，程序一旦启动，计算机便能自动地从头至尾执行完毕，并输出计算结果，由于在程序执行过程中不需要人进行干预，所以充分发挥了计算机高速运算的优点。

二、机器语言(Machine Language)

机器语言就是 2 进制编码的机器指令。计算机在运行过程中，首先将指令从内存贮器的指令代码区取至 CPU 的指令寄存器(IR)，然后，对指令代码进行译码，从而针对不同指令产生不同的操作控制信号，以完成指令所指定的功能。显然，只有这种指令代码符合特定计算机事先对代码的约定，计算机才能理解和执行之。用这种指令代码书写的程序叫机器语言程序。例如，若机器语言代码为 3E05H，则在 Z80 指令系统中，它的功能是将一个立即数 05H 送至 CPU 中的累加器 A。

用机器语言编写的程序，通常又称为目的程序(Object Program)，这种程序虽能为机器所理解并执行，但却存在着严重的缺点：用机器语言书写的程序既不便于理解、阅读和交流，且又是一件繁重的工作，极易出错。

三、汇编语言

为了便于理解和记忆指令，人们采用能帮助记忆的英文缩写符号(称为指令助记符)来代替指令的操作码。用指令助记符及符号地址所书写的指令叫汇编格式指令，而用汇编格式指令编写的程序叫汇编语言(Assembly Language Program)。

例如，Z80 的机器语言和对应的汇编语言如下：

机器语言	汇编语言
3E05H	LD A, 05H
0604H	ADD A, 04H
320010H	LD (0010H), A
76H	HALT
⋮	⋮

其中第1条是传送指令，操作码助记符为LD。这条指令的功能是将常数05H送入累加器A。

第2条为加法指令，操作码的助记符为ADD（英文中就是做加法的意思）。这条指令的功能是将常数04H与累加器A中的内容相加，且将结果保留在累加器A中。

第3条为传送指令，该指令的功能是把累加器A中的内容送到地址代码为0010H的存贮单元中去。

第4条为暂停指令，其功能是使程序到此暂时停止。

从此例可以看出，用汇编语言书写程序要比用机器语言书写程序方便得多，且汇编语言程序对执行过程有一定程度的描述，便于人们阅读和记忆。由于汇编语言程序与机器语言程序是一一对应的，所以汇编语言程序也是针对特定机器的程序。这在一定程度上有碍于程序在不同类型的机器之间进行交流。

虽然汇编语言程序为人们对程序的编写、阅读和交流带来了方便，然而机器只认识（理解）用机器语言编写的目的程序，因此必须在计算机执行程序之前，将汇编语言源程序（Source Program）翻译成机器语言的目的程序。这种翻译工作可以由人通过查表的办法进行，这种工作叫做人工汇编（或称代真）。人工汇编是一种单调、重复、繁重而效率极低的工作，它完全可以由电子计算机来进行。用电子计算机翻译的过程叫机器汇编。

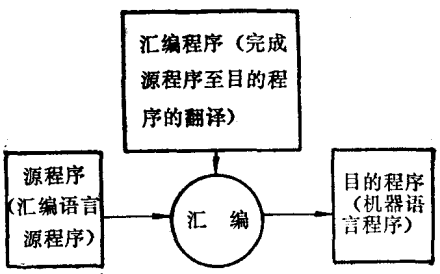

图1-2 汇编过程示意图

汇编过程示意图如图1-2所示。

四、高级语言(High Level Language)

汇编语言仍然是面向机器的语言，要求编程人员对特定机器的指令系统有详细深入的了解，而高级语言则是面向使用者（用户）的语言，它更接近于人们日常习惯的数学语言。它不要求编程人员对机器指令系统有深入地了解，一个用高级语言编写的程序可以在各种不同型号的机器上运行（只要配有该种语言的编译或解释程序）。高级语言的程序由语句组成，每个语句相当于若干条机器指令，因此语句的功能很强，并且用语句编写程序要容易得多。

目前通用的高级语言有：BASIC、FORTRAN、COBOL、PASCAL等等。

虽然高级语言使用方便，但对于实时控制的应用场合，却存在着明显的不足之处：与汇编语言程序相比，要多占用0.5～1倍的存贮单元，执行程序化费的时间要长0.5～2倍。汇编语言可以直接用输入输出指令通过低级接口与外设打交道，而高级语言却不那么方便。

使用汇编语言编写程序，能充分发挥机器硬件的作用，能高效地使用机器。汇编语言是编写计算机通用软件的基础，因而掌握汇编语言是十分重要的。

§1-4 Z80中央处理器

汇编语言是面向机器的语言，它要求熟悉某一机器的指令系统，要求了解有关机器硬件的结构，了解信息在机器中进行流动和加工的过程，即了解指令的执行过程。为此，我们对

Z80 CPU作一简要介绍。

目前，我国的8位微型机以Z80为CPU的占多数，而Z80单板机由于价格已降至500元以下，因而在实时控制、智能仪器中被广泛采用。在这些应用场合，要求使用者必须掌握Z80汇编语言程序设计的方法，并对Z80 CPU的内部结构，也应有所了解。

一、Z80 CPU的一般介绍

Z80是单片微处理器，功能较强，是8位微处理机中较好的一种，有人称它为微处理机的第3代产品。

Z80是采用MOS N沟道耗尽型负载工艺制造的大规模集成电路芯片。它是塑料封装的双列直插式芯片，具有40条引脚。引脚示意图如图1-3所示。

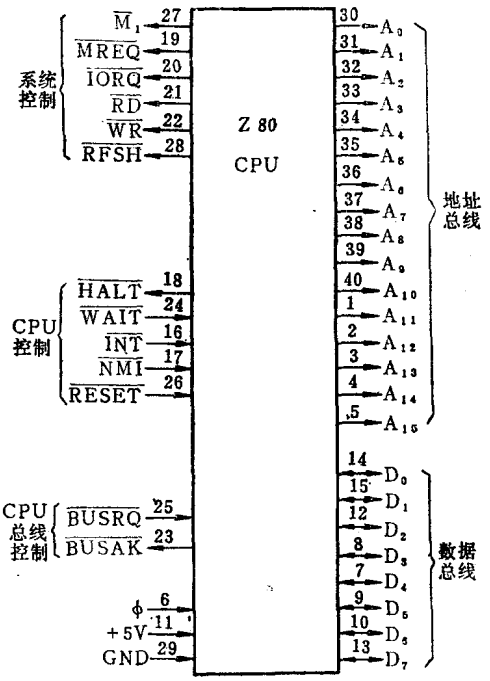

图1-3 Z80 CPU引脚示意图

（一）16根地址线

Z80 CPU的地址码为16位，可访问64K地址。因此，CPU芯片有16根地址线，编号为A_0～A_{15}，A_{15}为最高位，A_0为最低位。地址线为一组三态输出总线，高位代表1，低位代表0，高阻态时总线开路，可由别的设备占用总线。地址总线是CPU与存贮器、CPU与外设或者外设与外设、外设与存贮器进行数据交换时传送地址的通道。只有当总线信号代表外设地址时，才使用其低8位，即A_0～A_7，因此，它能直接选择的外设地址可达256个。

（二）8根数据线

Z80 CPU的字长为8位。因此，数据线为8根，其编号为D_0～D_7，D_7是最高位，D_0是最低位。数据线为一组三态双向总线，高电平代表1，低电平代表0，高阻态时总线开路，可供别的设备使用。数据总线是CPU与存贮器或CPU与外设进行交换信息的通道，其上的信号可双向传送。

（三）13根控制线

这13根控制线可进一步分为3类，即系统控制线、CPU控制线及总线控制线。

1. 系统控制线

系统一般指CPU以外的设备，如存贮器和外设。系统控制线就是对系统（存贮器、外设）控制的信号线，它由$\overline{M_1}$、$\overline{MREQ}$、$\overline{IORQ}$、$\overline{RD}$、$\overline{WR}$、$\overline{RFSH}$ 6根输出线组成，它们均是低电平有效的输出控制线，其作用是：

(1) $\overline{M_1}$表示CPU当前正在进行取指令的操作（从内存中读出指令的代码，在M_1机器周期完成）。

(2) $\overline{MREQ}$（存贮器请求）信号，三态输出，表示地址总线上的地址码是访问存贮器的地址，因此，这时机器进行的是对存贮器的读写操作。