

微型电子计算机教程

微型计算机 汇编语言及其应用

林卓然 编

广东科技出版社

内 容 简 介

本书以TRS-80和APPLE II两种微型机为学习机型,叙述了Z-80和6502汇编语言程序设计的基本概念、方法及其应用。主要内容包括:微处理器的基本结构和指令系统;程序设计的基本方法和技巧;输入/输出程序;调用系统子程序;上机操作等。书中注重分析这两种微型机系统的硬、软件结构,并列举了大量的实用例子,如汉字输出、EPROM读写器、数据采集、收款机等,供读者在使用中参考。

本书可作为微型机培训班的教材,也可供从事微型机应用的科技人员及大专院校、中学高年级学生阅读。

微型电子计算机教程 微型计算机汇编语言及其应用

林卓然 编

*

广东科技出版社出版

广东省新华书店发行

广东肇庆新华印刷厂印刷

787×1092毫米32开本 12印张 240,000字

1988年3月第1版 1988年3月第1次印刷

印数 1—3,700册

ISBN 7-5359-0280-4

TP·S 定价: 2.80元

前 言

汇编语言是微型计算机程序设计使用的主要语言之一。编写这本书，旨在帮助那些已经初步掌握微型机操作的同志尽快学习和使用这一重要的程序设计手段，以便进一步扩大微型机的应用范围和提高微型机的使用效率。

本书以TRS-80微型机为主要学习机型进行编写。编者对几年来国内外有关TRS-80微型机汇编语言的资料进行归纳和整理，同时参考了有关书籍和收集了大量实用程序，因此本书是学习TRS-80微型机汇编语言的较为系统的参考书。书中还以较大篇幅介绍了APPLE II微型机的汇编语言，这对于使用6502微处理器的读者也有一定参考价值。

本书是继《TRS-80微型计算机BASIC II语言与磁盘操作系统》一书之后编写的，这两本书的内容虽然不同，但本书的某些章节会用到前一本书所介绍的一些基本概念。对于初学者来说，建议最好先将前一本书的主要章节阅读一遍，待掌握了一定的BASIC语言编程知识和操作方法之后，再来学习这本书。

由于编者水平有限，书中一定存在不少缺点和错误，敬请读者批评指正。

1986年8月 于广州康乐

目 录

第一部分 TRS—80微型机汇编语言

第一章 初步认识汇编语言	(1)
§ 1.1 微型机的程序设计语言.....	(1)
§ 1.2 汇编语言程序的基本特点.....	(5)
§ 1.3 怎样上机操作.....	(7)
第二章 微型计算机系统的基本结构	(11)
§ 2.1 微型机系统结构.....	(11)
§ 2.2 Z-80CPU的结构.....	(15)
§ 2.3 CPU寄存器.....	(18)
第三章 Z-80的指令系统	(25)
§ 3.1 数据、地址和指令.....	(25)
§ 3.2 指令的长度和执行时间.....	(28)
§ 3.3 Z-80的指令系统.....	(31)
§ 3.4 寻址方式.....	(56)
习题.....	(62)
第四章 汇编语言的基本成分	(65)
§ 4.1 常数和字符串.....	(65)
§ 4.2 标号.....	(65)
§ 4.3 表达式.....	(66)
§ 4.4 伪指令.....	(67)
第五章 汇编语言的程序设计	(70)
§ 5.1 程序设计的步骤.....	(70)
§ 5.2 简单程序.....	(71)

§ 5.3	分支程序	(79)
§ 5.4	循环程序	(88)
§ 5.5	子程序	(107)
§ 5.6	优化程序	(122)
§ 5.7	算术运算程序例	(127)
习题		(138)
第六章 编写输入/输出的程序		(144)
§ 6.1	概述	(144)
§ 6.2	程序控制输入/输出传送	(147)
§ 6.3	无条件传送	(148)
§ 6.4	条件传送	(150)
§ 6.5	中断传送	(152)
第七章 调用 T R S-80 微型机的系统子程序		(165)
§ 7.1	调用 ROM 中标准子程序	(165)
§ 7.2	调用存取磁盘文件的子程序	(166)
第八章 实例介绍		(179)
§ 8.1	用计算机演奏乐曲	(179)
§ 8.2	实现汉字输出	(188)
§ 8.3	修改系统程序	(197)
§ 8.4	EPROM 读写器	(201)
§ 8.5	数据采集系统	(211)
§ 8.6	收款机简介	(218)
第九章 汇编语言的操作		(228)
§ 9.1	概述	(228)
§ 9.2	编辑/汇编命令	(229)
§ 9.3	控制汇编列表的命令	(236)
§ 9.4	使用磁带上的编辑/汇编程序	(236)

第十章 调试机器语言程序.....	(238)
第十一章 其他有关操作.....	(244)
§ 11.1 反汇编操作.....	(244)
§ 11.2 使用SUPERZAP程序.....	(247)
§ 11.3 新DOS80的几个操作命令.....	(251)

第二部分 APPLE II 微型机汇编语言

第十二章 概述.....	(253)
§ 12.1 初步认识.....	(253)
§ 12.2 6502微处理器简介.....	(256)
§ 12.3 指令的分类.....	(259)
§ 12.4 6502的寻址方式.....	(263)
第十三章 程序设计举例.....	(270)
§ 13.1 APPLE II 微型机汇编语言的一些规定	(270)
§ 13.2 简单程序.....	(272)
§ 13.3 分支程序.....	(276)
§ 13.4 循环程序.....	(278)
§ 13.5 子程序.....	(285)
习题.....	(289)
第十四章 BASIC程序和机器语言子程序的联用	(292)
§ 14.1 建立机器语言子程序.....	(292)
§ 14.2 调用机器语言子程序.....	(295)
§ 14.3 调用BASIC程序变量.....	(300)
第十五章 APPLE II 微型机编辑/汇编的命令	(307)

第十六章 使用监控程序.....	(312)
第十七章 使用小型汇编程序.....	(324)
附录.....	(328)
一、字符码.....	(328)
二、TRS-80微型机的内存贮器分配图.....	(329)
三、Z-80指令系统表.....	(333)
四、APPLE II机的内存贮器分配图.....	(361)
五、6502指令系统表.....	(363)

第一部分 TRS-80微型机汇编语言

第一章 初步认识汇编语言

§ 1.1 微型机的程序设计语言

微型计算机所用的程序设计语言也和中小型计算机类似。它基本上可分成三类：一类是完全面向机器的机器语言；一类是比机器语言较容易为人理解的，同时又非常接近机器语言的符号语言，称为汇编语言；还有一类是面向问题处理过程的语言，亦即高级语言。

一、机器语言

每台微型机都有自己的指令系统，机器就靠这些指令的组合来完成某一特定的操作。指令以二进制形式编码存放在存储器中，需要时再行调出。由这种二进制形式编码的指令就称为机器语言。从整个计算机发展的历史来看，这是最早使用的低级语言。使用机器语言编写的程序，可直接在计算机上执行，但由于这样的程序没有明显的特征，因此不好记忆和掌握，易出错，出了错又不便于修改。下面列出一个能在TRS-80微型机显示屏的中心位置(地址为3 E20H，H表示十六进制数)显示出字符“1”的机器语言程序(见表1.1)。

表 1.1

一个机器语言程序

存储器地址 (用十六进制表示)	内容 (指令代码)	
	(用二进制表示)	(用十六进制表示)
7D01	00111110	3E
7D02	00110001	31
7D03	00110010	32
7D04	00100000	20
7D05	00111110	3E
7D06	11000011	C3
7D07	00000001	01
7D08	01111101	7D

显然，初学者是很难理解上述程序包含什么内容的，比较熟练的程序员也要根据指令编码表逐条查对才能理解它。

由于机器语言存在上述许多不便之处，故一般不用它编制程序。

二、汇编语言

汇编语言是一种用记忆符号来替代机器语言指令代码的程序设计语言。如用记忆符ADD代替加法指令代码11000~1110，用SUB (Subtract) 代替减法指令代码11010110等等，这样就比较容易记住指令的含意。

汇编语言还允许使用标号来代表指令所涉及的地址或操作数。例如，可用标号LOOP来代表转移地址，把转移指令写成JP LOOP，其意思是转移 (Jump) 至 LOOP 指定的地址去执行指令。

前述的机器语言程序，若改用汇编语言来编写，则有

```
100      ORG 7D01H 表示从7D01H开始存放程序
```

110	LOOP: LD	A, 31H	把1的ASCII码存入累加器A
120	LD	(3E20H), A	把A中内容送入显示屏中心位置
130	JP	LOOP	转移至LOOP去执行
140	END	7D01H	表示源程序结束。并确定7D01H为将来执行程序的入口地址

这样，整个程序就比较容易看懂。

但这样写成的程序，计算机是无法直接执行的，还存在一个“翻译”问题。把汇编语言程序翻译为机器语言程序的过程称为“汇编”。用汇编语言写的程序称为“源程序”，汇编产生的机器语言程序称为“目标程序”。

汇编的方法有两种：

第一种方法是人工汇编，即由程序员直接查指令表，并把源程序中的每个指令记忆符和标号代换为它们所对应的代码。显然，这个工作是非常烦琐和容易出错的。

第二种方法是机器汇编。这是绝大多数微型机采用的方法。它利用一种软件——“汇编程序”来完成这种烦琐的重复性工作。汇编程序能把汇编语言源程序汇编为指令代码形式的目标程序。图1.1描述了这个工作过程。

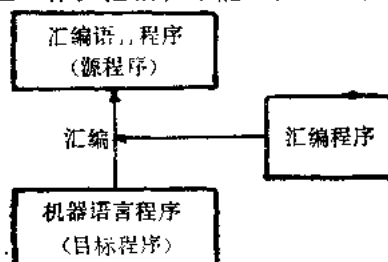


图1.1 机器汇编过程

汇编语言与机器语言相比，它的优点是非常明显的。现在几乎每一种微型机系统都配置有自己的汇编语言。

三、高级语言

汇编语言比机器语言前进了一大步，但是它和机器语言一样是与具体的机器紧密相联系的，要使用它就必须熟悉机器的指令系统，掌握必要的硬件知识。对于不同种类的微型机，针对同一问题所写的汇编语言程序是不能通用的。其次，一个汇编语句通常仅对应一条机器指令，因此即使完成一件不大的工作，也需要用大量的汇编语句，使用仍不便，不易掌握。

为了避免汇编语言的这些缺点，就必须使用高级语言。目前在微型机系统中常用的高级语言有 BASIC、FORTRAN、COBOL 等。高级语言不依赖或很少依赖于机器，而且更接近于人们的书写习惯，因此通用性强，容易学习和运用。它们的一个语句往往相当于许多条机器指令，编出来的程序更加简单明了，便于调试和修改。

不过高级语言也有一些不足之处，这就是：采用高级语言编写的源程序，要花费较多的时间进行语言处理；由于它的通用性，所以就不能充分发挥每个机器的特点，机器的效率会下降；高级语言对于微型机应用中所需要的接口技术等并不十分有效。

汇编语言虽然比高级语言更依赖于机器，但它却能充分发挥具体微型机的特长和灵活性，从而就能编制出效率较高的程序。许多微型机软件（包括系统软件）都是用汇编语言写成的。此外，汇编语言程序能准确地掌握执行时间，适用于实时控制。

因此，虽然有了高级语言，但汇编语言仍不失为微型计算机的主要程序设计手段之一。

§ 1.2 汇编语言程序的基本特点

汇编语言是一种面向机器的程序设计语言，它与机器结构有关，因此不同的机器所配的汇编语言各不相同。不过，各种八位微型机所采用的汇编语言规则却有很多相似之处，掌握了其中的一种之后，对其他微型机的汇编语言也就能够触类旁通了。

在上一节里，我们已经遇到了一个用TRS-80 微型机汇编语言编写的程序，下面就来介绍这样的程序有哪些基本特点。

TRS-80汇编语言程序有以下几个基本特点：

1. 一个汇编语言程序是由若干个汇编行组成的。上述程序有5个汇编行。每个汇编行只允许写一个汇编语句。

2. 汇编语句由四个部分组成，每个部分称为一个字段，其结构如下：

标号：	操作码	操作数	；注释内容
-----	-----	-----	-------

各字段之间用分隔符分开。标号之后用冒号或空格分开，操作码和操作数之间用空格分开，注释内容之前需加上分号，当有两个操作数时，中间用逗号“，”分开。

下面是几个典型的汇编语句：

标号	操作码	操作数	注释内容
	LD	A, (7000H)	GET LENGTH
	RLA		
DONE:	ADD	A, B	

操作码和操作数是构成汇编语言指令的两个部分。汇编语言指令分为两类：一类是微处理器指令（简称指令），如 LD A, 31H、JP LOOP等，在汇编时，这类指令将被翻译成一对应的机器码；另一类是汇编程序的伪指令，如 ORG 7D01H、END 7D01H等。这些伪指令是为在汇编操作时完成一些指定任务而专设的，汇编时伪指令并不被译成机器码。

(1) 标号字段

标号是为语句定义的名称（或称标识符），它表示汇编语言指令的符号地址。

每个语句不一定需要标号，只有那些被其他语句引用的语句，才需要赋予标号。当无标号时，语句的标号字段为空白。

(2) 操作码字段

这是语句中唯一不可省略的部分，它规定了语句的操作性质。操作码由汇编语言指令的记忆符（通常是汇编语言指令的英文缩写形式）来表示。例如：

LD——load 装载(传送)

JP——jump 跳至

ORG——origin 开始

END——end 结束

(3) 操作数字段

这是语句中最复杂的部分，它的个数及形式随汇编语言指令而不同。例如：

LD (7DFEH), A 具有用逗号分开的两项

JP START 一项

CPL 无操作数字段

(4) 注释字段

这个字段仅对语句做些说明，以便有助于使用者阅读程序。它不影响源程序的汇编结果，同时也不被翻译成任何机器码。

在汇编行中，为了便于标识和有助于以后的上机操作，还可以在该行的开头加上一个行号，正如上一节所列出的程序一样。所取的行号是按汇编语句顺序排列的，一般从100开始，增量为10。

3. 汇编语言程序中用伪指令“ORG”和“END”来分别表示程序的开头和结束。ORG后面的地址是表示从该地址开始存放目标程序。END可以后缀一个地址，它表示将来执行目标程序时从该地址启动。

§ 1.3 怎样上机操作

编好了程序，就可以上机操作，其操作步骤如下：

1. 调用编辑/汇编程序

TRS-80机的编辑/汇编程序是一个系统软件，它能供使用者进行汇编语言的编辑操作和汇编操作。

在DOS操作状态下^①，打入EDTASM和按ENTER键，即

```
DOS READY
EDTASM      ENTER
TRS-80  EDITOR/ASSEMBLER  1.1
```

*~

^①除了有特别说明，本书所介绍的所有操作都是在TRS-80 NEW DOS 80（新DOS 80）或NEW DOS+（新DOS）的状态下进行的。

这一显示说明计算机处于编辑/汇编程序的管理之下。

2. 输入源程序

进行如下的操作:

```
*L100, 10          ENTER
00100→          ORG→      7D01H          ENTER
00110LOOP:→      LD→      A, 31H          ENTER
00120→          LD→      (3E20H), A      ENTER
00130→          JP→      LOOP          ENTER
00140→          END→      7D01H          ENTER
00150BREAK
```

*-

说明: (1)在上述操作过程中,除了每行开头的行号之外,其他都是使用者打入的。

ENTER、BREAK和→都是TRS-80机的特殊功能键。

(2)→键能使显示屏的光标以定点位置移动(见BASIC I语言的特殊功能键)。在字段之间打入→键,可起到加入空格分隔符的作用,另外,它也能使各字段列对齐,方便阅读。

3. 对源程序进行汇编、列表,并将所得的目标程序存入磁盘(或磁带)。

在以上状态下,进行如下操作:

```
*A          ENTER
OBJECT FILE TO DISK OR TAPE? RE
PLY "R" OR "T".
D          ENTER
OBJECT FILESPEC?
DIS/CMD:0          ENTER
```

***** FILE NON-EXISTENT. REPLY "C" TO CREATE IT

C ENTER

```

7D01          00100          ORG 7D01H
7D01 3E31     00110  LOOP: LD  A,31H
7D03 32203E   00120          LD  (3E20H),A
7D06 C3017D   00130          JP  LOOP
7D01          00140          END 7D01H
00000 TOTAL  ERRORS
LOOP  7D01  00110  00130

```

*-

在上述程序清单中，第一列是存贮地址。其中第一个数7D01H表示程序的首地址，第二个数7D01H表示110行指令的开始存放地址。这条指令占用两个字节，故第二条指令(120行)的存放地址从7D03H开始，以下依此类推。最后的7D-01H是将来执行该机器语言程序的入口地址。

第二列为指令代码，例如第二条指令LD (3E20H), A，它的第一个字节为操作码32H，第二、三字节分别为操作数的低位地址20H和高位地址3EH，其余指令代码依此类推。

第三列以后为源程序清单。

程序清单的下面是“00000 TOTAL ERRORS”，它用五位数字表示错误指令的条数。现在是0，说明汇编结果全部正确。

最后是标号表，它列出了源程序中所用的标号及其代表的地址和出现的行号。

4. 如何执行机器语言程序

使用汇编语言编写源程序，以及对源程序进行汇编，这是得到机器语言程序的一种重要手段。但是我们的最终目的是从执行机器语言程序中得到结果。

对于上述建立在磁盘中的DIS/CMD:0机器语言程序文件，可采用如下方法执行：

在DOS操作状态下，打入该文件标识符 和 按 ENTER 键，即

DOS READY

DIS/CMD:0 ENTER

此时，磁盘操作系统就把该程序文件装入内存，存贮区域为地址7D01H~7D08H，给程序计数器PC赋以7D01H 并从该地址开始执行。在一般情况下，机器是按存贮地址的顺序来取出和执行指令的，但若遇到控制转移指令，则会转移到指定的地址去执行。对于上述程序，其执行的情况如下：先执行从地址7D01H开始的指令LD A, 31H，即将字符“1”的ASCII码31H放入累加器 A（A是CPU内部的一个部件）中，接着执行地址 7D03H 开始的指令 LD (3E20H)，A，即把A中的内容31H放入 3E20H 的地址里，此时就会在显示器中央位置显示出字符“1”，最后执行地址 7D06H开始的指令 JP 7D01H，此指令的含意是跳至地址 7D01H去执行，这样又重复以上工作过程。