# 5.直接程序设计

5.1汇编语言程序的功能与编制过程

汇编程序的主要功能：  
1. 检查源程序

2. 测出源程序中的语法错误，并给出出错信息。

3. 产生源程序的目标程序，并可给出列表文件（同时列出汇编语言和机器语言的文件，成为LST文件）

4. 展开宏指令

编制汇编语言程序的过程

**1. 分析题意,确定算法,找出合适的数据结构.**

**2. 根据算法,画出程序框图.**

**3. 根据框图编写程序.**

**4. 上机调试程序.**

**常用程序有顺序,循环,选择,和子程序结构四种结构形式.**

**实验一、调试工具DEBUG的应用**

**习题：（见总体习题解答文档第五章）**

1.简述判断程汇编语言程序序质量的标准。

答：

一个良好的程序应符合以下标准：

（1）程序可读性强。

（2）程序执行的时间短（效率高）。

（3）程序所占用的内存少。

（4）程序的语句行数少。

2.……

5.2汇编语言程序的组成

汇编语言程序的语句的组成可分为指令、伪操作和宏指令。

**指令**：每一条指令都对应一个CPU操作。

**伪操作**：又称作伪指令。不像机器指令在程序运行期间由计算机来执行，是在汇编程序对源程序汇编期间由汇编程序处理的操作。它是用于告诉汇编程序如何进行汇编的指令。既不会控制机器的操作也不会被汇编成机器代码，只能为汇编程序所识别并指导汇编如何进行。伪操作可以完成如数据定义、分配储存区、指示程序结束等功能。

**宏指令**：由用户按照宏定义格式编写的一段程序。

5.3汇编语言程序格式

**汇编语言程序格式**

汇编语言源程序中的每个语句可以由四项组成，格式如下：

[名字项] 操作项 操作数项 [;注释项]

**5.3.1 名字项**

名字项是一个符号。源程序使用下列字符来表示名字：字母A~Z，数字0~9，专用字符?、·、@、$、—。除数字外，所有字符都可以放在源语句的第一个位置。名字中如果用到·则其必须为第一个字符，可以用很多字符来说明文字，但只有前31个字符能被汇编程序所识别。

名字项可以是标号或者变量。它们都用于表示本语句的符号地址，是可有可无的，只有当需要使用符号地址来访问该语句时才需要出现。

**标号** 标号在代码段中定义，后面跟冒号:，它也可以用LABEL或EQU伪操作来定义。标号经常在转移指令或CALL指令的操作数字段出现，用以表示转向地址。

**标号有三种属性**：

**段属性**：定义标号的段起始地址，此值必须在一个段寄存器中，而标号的段则总是在CS寄存器中。

**偏移属性**：标号的偏移地址是16位无符号数，它代表从段起始地址到定义标号的位置之间的字节数。

**类型属性**：用来指出该标号是在本段内引用还是在其他段中引用的。如在段内引用，则称为NEAR，指针长度为2字节；如在段外引用，则称为FAR，指针长度为4字节。

**变量** 变量在除代码段以外的其他段中定义，后面不跟冒号。也可以用LABEL或EQU伪操作来定义。变量经常在操作数字段出现，也有段、偏移和类型三种属性。

**段属性**：定义变量的起始地址，此值必须在一个段寄存器中。

**偏移属性**：变量的偏移地址是16位无符号数，它代表从段的起始地址到定义变量的位置之间的字节数。在当前段内给出变量的偏移值等于当前地址计数器的值，当前地址计数器的值可以用$来表示。

**类型属性**：变量的类型属性定义该变量所保留的字节数。

在程序中同样的标号或者变量的定义只允许出现一次，否则汇编程序会提示出错。

**5.3.2 操作项**

操作项可以是指令、伪操作或宏指令的助记符。对于指令，汇编程序会将其翻译为机器语言指令。对于伪操作，汇编程序将根据其所要求的功能进行处理。对于宏指令，则将根据其定义展开。

**5.3.3 操作数项**

操作数项由一个或多个表达式组成，多个操作数项之间一般用逗号分开。对于指令，操作数项一般给出操作数地址，可能有一个或两个，也可能一个也没有。对于伪操作或宏指令则给出所需要的参数。

操作数项可以是常数、寄存器、标号或由表达式组成。

**表达式**

表达式是常数、寄存器、标号、变量与一些操作符相组合的序列，可以有数字表达式和地址表达式两种。在汇编期间，汇编程序会按照一定的优先规则对表达式进行计算后得到一个数值或一个地址。

**表达式中常用的操作符**

**算术操作符**

算术操作符有+、-、\*、/和MOD。可以用于数字表达式或地址表达式中，但用于地址表达式时，只有当其结果有明确的物理意义时其结果才是有效的。

**逻辑操作符**

逻辑操作符有AND、OR、XOR和NOT，逻辑操作符是按位操作的，只能用于数字表达式中。

视频讲解

**关系操作符**

关系操作符有EQ（相等）、NE（不等）、LT（小于）、GT（大于）、LE（小于或等于）、GE（大于或等于）6种。关系操作符的两个操作数必须都是数字或是同一段内的两个存储器地址。计算结果为逻辑值；结果为真表示为0FFFFH；结果为假表示为0.

**数值回送操作符**

有TYPE、LENGTH、SIZE、OFFSET、SEG五种。这些操作符可以把一些特征或存储器地址的一部分作为数值回送。

**TYPE**

格式为

TYPE 变量或标号

如果为变量，则汇编程序将回送该变量的以字节数表示的类型，DB为1，DW为2，DD为4，DQ为8，DT为10；如果是标号，则汇编程序将送回代表该标号类型的数值：NEAR为-1，FAR为-2。

**LENGTH**

格式为

LENGTH 变量

对于变量中使用DUP的情况，汇编程序会回送分配给该变量的单元数。而对于其他情况则回送1。

**SIZE**

格式为

SIZE 变量

汇编程序会回送分配给该变量的字节数。但是，此值为LENGTH值和TYPE值的乘积。

**OFFSET**

格式为

OFFSET 变量或标号

汇编程序会回送变量或者标号的偏移地址值。

**SEG**

格式为

SEG 变量或标号

汇编程序会回送变量或者标号的段地址值。

视频讲解使用

**属性操作符**

有PTR、段操作符、SHORT、THIS、HIGH和LOW六种。

**PTR**

格式为 类型 PTR 表达式

PTR用来建立一个符号地址，但本身并不分配存储器，只是用来给已分配的储存地址赋予另一种属性，使该地址具有另一种类型。格式中的类型字段表示所赋予的新的属性类型，而表达式字段则是被取代类型的符号地址。

**段操作符**

用来表示一个标量、变量或地址表达式的段属性。

**SHORT**

用来修饰JMP指令中转向地址的属性，指出转向地址是在下一条指令的±127个字节范围内。

**THIS**

格式为THIS 类型或距离

它可以像PTR一样建立一个指定类型（BYTE、WORD或DWORD）的或指定距离（NEAR或FAR）的地址操作数。该操作数的段地址和偏移地址与下一个存储单元地址相同。

**HIGH和LOW**

称为字节分离操作符，它接收一个数或地址表达式，HIGH取其高位字节，LOW取其低位字节。

**操作符计算的优先级别**

1在圆括号中的项，在方括号中的项，结构变量（变量、字段），然后是LENGTH、SIZE、WIDTH和MASK。

2名：（段取代）。

3PTR，OFFEST，SEG，TYPE，THIS及段操作符。

4HIGH和LOW。

5乘法和除法：\*，/，MOD，SHL，SHR

6加法和减法：+，-

7关系操作：EQ，NE，LT，LE，GT，GE

8逻辑：NOT

9逻辑：AND

10逻辑：OR，XOR

11 SHORT

**5.3.4 注释项**

注释项用来说明一段程序或一条或几条指令的功能，是可有可无的。但是对于汇编语言，注释项的作用十分明显，可以使程序易于被读懂，因此编制汇编语言程序必须写好注释。注释应该写出被注释的指令或程序的功能或作用而不是只写指令的动作。

习题（见总体习题解答文档）

5.4 伪操作

**5.4.1数据定义及储存器分配伪操作**

格式：[变量] 助记符 操作数,…,操作数 [;注释]

**变量字段** 可有可无，由符号地址表示，作用与指令语句前的标号相同但是后边不加冒号。如果语句中有变量则汇编程序使其记为第一个字节的偏移地址。

**注释字段** 用于说明该伪操作的功能，也是可有可无的。

**助记符字段** 用于说明所用伪操作的助记符，常用的种类有：

DB伪操作：用来定义字节，其后的每个操作数都占有一个字节。

DW伪操作：用来定义字，其后的每个操作数都占有一个字（低位字节存放于第一个字节地址，高位字节存放于第二个字节地址）。

DD伪操作：用来定义双字，其后的每个操作数占有两个字。

DQ伪操作：用来定义四个字，其后的每个操作数占有四个字。

DT伪操作：用来定义十个字节，形成压缩的BCD码。

这些伪操作可以把其后的数据存入指定的存储单元，或者只分配存储器空间而并不存入确定的数值。DW和DD伪操作可以存储偏移地址（DW）或完整的地址（DD）。使用DD伪操作存入地址时，第一个字为偏移地址，第二个字是段地址。

**操作数** 数据定义及储存器分配伪操作后的操作数可以是常数、表达式或者字符串。操作数？可以保留存储空间，但不存入数据。

操作数字段可以使用复制操作符DUP复制操作数。格式为：

重复次数 DUP (操作数,…,操作数)

重复次数可以为常数或者一个表达式，值应该是一个正整数，用来指定括号中操作数的重复次数。DUP复制操作数可以嵌套使用。

操作数字段中的变量或标号可以使用表达式，如：

变量±常数表达式 标号±常数表达式

在这种情况下，经过汇编后存储器中应该存入表达式的值。

**变量的类型属性**

在数据定义伪操作前的变量的值是该伪操作中的第一个数据项在当前段内的第一个字节的偏移地址。而变量的类型属性则用于表示语句中每一个数据项的长度（以字节单位表示），变量表达式的属性和变量是相同的。根据这个隐藏的类型属性，汇编程序可以确定某些指令是字指令还是字节指令。

可以利用PTR属性操作符指定操作数的类型属性，它优先于隐含的变量类型属性。其格式为：

类型 PTR 变量±常数表达式

类型可以为BYTE（字节），WORD（字）或DWORD（双字）。利用PTR属性操作符可以使同一个变量可以具有不同的类型属性。除了利用PTR属性操作符外还可以使用LABEL伪操作来定义。格式为：

变量或标号的名字 LABEL 类型

对于数据项，变量的类型可以为BYTE（字节），WORD（字）或DWORD（双字）。对于可执行的代码，标号的类型可以为NEAR或者FAR。

视频讲解数据定义及存储器分配伪操作的使用，包括DB\DW区别，？，DUP等使用，操作数类型匹配问题

**5.4.2表达式赋值伪操作EQU**

利用赋值伪操作可以给表达式赋予一个名字，在程序中多次出现该表达式时可以利用表达式的名字进行代替。其格式如下：

表达式名 EQU 表达式

格式中的表达式可以是任何有效的操作数格式、常数值的表达式和任何有效的助记符。如果EQU语句表达式中如果有变量或标号的表达式，则必须在该语句前给出变量或标号的定义。

=伪操作也可以作为赋值伪操作使用。与EQU不同的是，EQU伪操作中的表达式名不允许重复定义而=伪操作允许重复定义。

**5.4.3段定义伪操作**

存储器的物理地址是由段地址和偏移地址组合而成。汇编程序在把源程序转换为目标程序时，必须确定标号和变量的偏移地址，并且需要把有关信息通过目标模块传送给连接程序，以便连接程序把不同的段和模块连接在一起形成一个可执行程序。为此需要使用段定义伪操作，其格式如下：

段名 SEGMENT [定位类型][组合类型][‘类别’]

…

…

…

段名 ENDS

对于数据段、附加段和堆栈段来说，中间部分一般是存储单元的定义、分配等伪操作。对于代码段则是指令及伪操作。

**定位类型** 用于告诉汇编程序如何确定逻辑段的地址边界，它可以是：

PARA 指定段的起始地址必须从小段边界开始，即段地址的最低的16进制数位必须为0，在不填写定位类型时为默认的定位类型。

BYTE 该段可以从任何地址开始

WORD 该段必须从字的边界开始，即段地址必须为偶数

PAGE 该段必须从页（256字节定义为1页）的边界开始，即段地址的最低两个16进制数位必须为0。

**组合类型** 主要用于多模块程序中。组合类型用于告诉汇编程序，当一个逻辑段装入储存器时，它与其他段如何进行组合。组合类型可以是：

NONE 本段与其他逻辑段不组合，对不同程序模块中的逻辑段，即使有相同的段名，也分别作为不同的逻辑段装入内存而不组合，在不填写组合类型时为默认的组合类型。

PUBLIC 该段连接时将与有相同名字的其他分段连接在一起。连接次序由连接命令指定。

STACK 指定该段在运行时为堆栈段的一部分，堆栈段具有相同的段名时组合在一起。

COMMON 对于不同程序模块中用COMMON说明的同名逻辑段，连接时装入同一个地址，即连接时该段与其他同名分段具有相同的起始地址，会产生覆盖。连接长度为所有段中最长段的长度，重叠部分的内容为最后一个段的内容。

MEMORY 指定该段将分配在所有其他连接在一起的段的前面，即地址的最高位。如果遇到多个MEMORY段，将遇到的第一个段作为MEMORY段，其余为COMMON段。

AT 表达式 使段的起始地点为表达式所计算出来的16位段地址，不能用来指定代码段。

**类别** 由单引号括起来的字符串，用于组成段组的名字。当几个程序模块进行连接时，将具有相同别名的逻辑段装入连续的内存区域，类名相同的辑段按先后顺序排列，没有类名的段与其他无类别段一起连续装入内存。

**5.4.4 设定段寄存器伪指令ASSUME**

为了指明段和段寄存器之间的关系，需要使用设定段寄存器伪指令ASSUME，其格式为

ASSUME 段寄存器名:段名

该伪指令用于向汇编程序说明定义的逻辑段属于何种类型的逻辑段。其中段寄存器名必须是CS、DS、ES和SS中的一个，而段名必须是由SEGMENT定义的段中的段名。

ASSUME NOTHING可以取消之前由ASSUME指定的段寄存器。

**5.4.5 程序开始和结束伪操作**

在程序的开始可以用NAME或TITLE为模块取名。

NAME伪操作的格式是：

NAME 模块名

汇编程序会将给出的模块名作为该模块的名字。如果程序中没有NAME伪操作，则也可以使用TITLE伪操作，其格式为：

TITLE 标题

TITLE伪操作可指定每一页上打印的标题。同时在没有使用NAME伪操作时，汇编程序会使用标题的前六个字符作为模块名。标题中最多可有60个字符。如果程序中既无NAME伪操作也无TITLE伪操作，则会使用源文件名作为模块名，所以NAME以及TITLE伪操作并不是必要的。但一般经常使用TITLE，以便在列表文件中能打印出标题。

**表示源程序结束的伪操作**的格式为：

END [标号]

其中标号指示程序开始执行的起始地址。如果有多个程序模块相连接，则只有主程序要使用标号，子程序模块只用END而不必指定标号。

**5.4.6 地址计数器和对准伪操作**

**EVEN伪操作**

一个字的地址最好从偶地址开始，所以为保证一个字数组从偶地址开始，可以在前面使用EVEN伪操作来达到这一目的。

**ORG伪操作**

格式为 ORG 常数表达式

如果常数表达式的值为n，则ORG伪操作可以使下一个字节的地址成为常数表达式的值n。在汇编程序对源程序汇编的过程中，会使用地址计数器来保存当前正在汇编的指令的地址。地址计数器的值可以用$表示，汇编语言允许用户直接使用$来引用地址计数器的值。因此

ORG $+8

可以表示跳过八个字节的存储区

**$的作用**

在指令和伪操作中可以直接使用$来表示地址计数器的值，如

JNE $+6

则转向地址是JNE指令的首地址加上6。

当$用在指令中时，它表示本条指令的第一个字节的地址。当其用于跳转指令时，其计算结果必须为另一条指令的首地址，否则汇编程序将会指示出错信息。当$用在伪操作的参数字段时，表示的是地址计数器的当前值。

视频讲解$，ORG等的使用

**5.4.7 基数控制伪操作**

汇编程序默认的数为十进制数，除非专门指定，汇编程序把程序中出现的数均看作十进制数，当使用其他基数表示常数时，需要专门给予标记，如：

**二进制数**：由一串0和1组成的数，其后跟以字母B

**十进制数**：由0~9的数字组成的数。一般情况下后面不必加上标记，在指定其他基数的情况下，后面可跟字母D

**十六进制数**：由0~9及A~F组成的数，后面跟以字母H。这个数的第一个字符必须为0~9，如果第一个字符为A~F时，应在其前加上数字0

**八进制数**：由数字0~7组成的数，后面可跟字母O或Q

**RADIX伪操作** 可以把默认的技术改变为2~16范围内的任何基数。

格式为

RADIX 表达式

其中表达式用来表示基数值（用十进制数表示）。需要注意的是在使用RADIX 16把基数定位16进制后，十进制数后面都应跟字母D，如果某个十六进制数的末字符为D，则应在其后跟字母H避免混淆。

**字符串** 可以看成串常数，可以用单引号或双引号把字符串放在其中，得到的是字符串的ASCII码值。