第3章 8086CPU指令系统

3.1 汇编语言指令

1. 基本概念

（1）机器语言

机器（CPU）能直接识别的一种二进制代码语言。

CPU能识别的一组二进制代码就是一条指令。

（2）机器语言程序

采用机器语言编写的程序，即二进制代码程序。

优点：程序送入计算机后，CPU可以直接执行。

缺点：不易书写，不易检查，编写程序十分困难。

（3）汇编语言

采用一组字母、数字和符号来代替一条二进制代码指令，这种表示指令的符号称为助记符，用一组符号来代替一条指令并编写程序时采用的语言，称为汇编语言。

目的：为了克服机器语言不易书写、记忆复杂等缺点。

第3章 8086CPU指令系统

汇编语言与高级语言区别：

例如：语句 X=A+B实现加法功能

- 在高级语言中，只要给变量A和B赋一确定值，此加法就可以实现。
- 在汇编语言中，程序必须指出A、B存放在何处，相加后的结果又存放在何处，然后才能实现这一加法运算。
- 汇编语言通过程序告诉计算机做什么和如何做，语言的实现更加具体。因此，该语言与计算机（处理器）紧密相关，从而要求学习和使用汇编语言需要对处理器的结构有更加深入的了解。

（4）汇编语言程序

- 用汇编语言编写的程序称为汇编语言程序，或者称为汇编语言源程序。

  这种编程方法称为汇编语言程序设计。

- 汇编语言源程序名必须为文件名.ASM。

  优点：比一串二进制代码清晰，书写容易，记忆方便。

  缺点：CPU不能直接执行，必须经过汇编，将其翻译成机器语言格式后，CPU才能执行。

（5）汇编

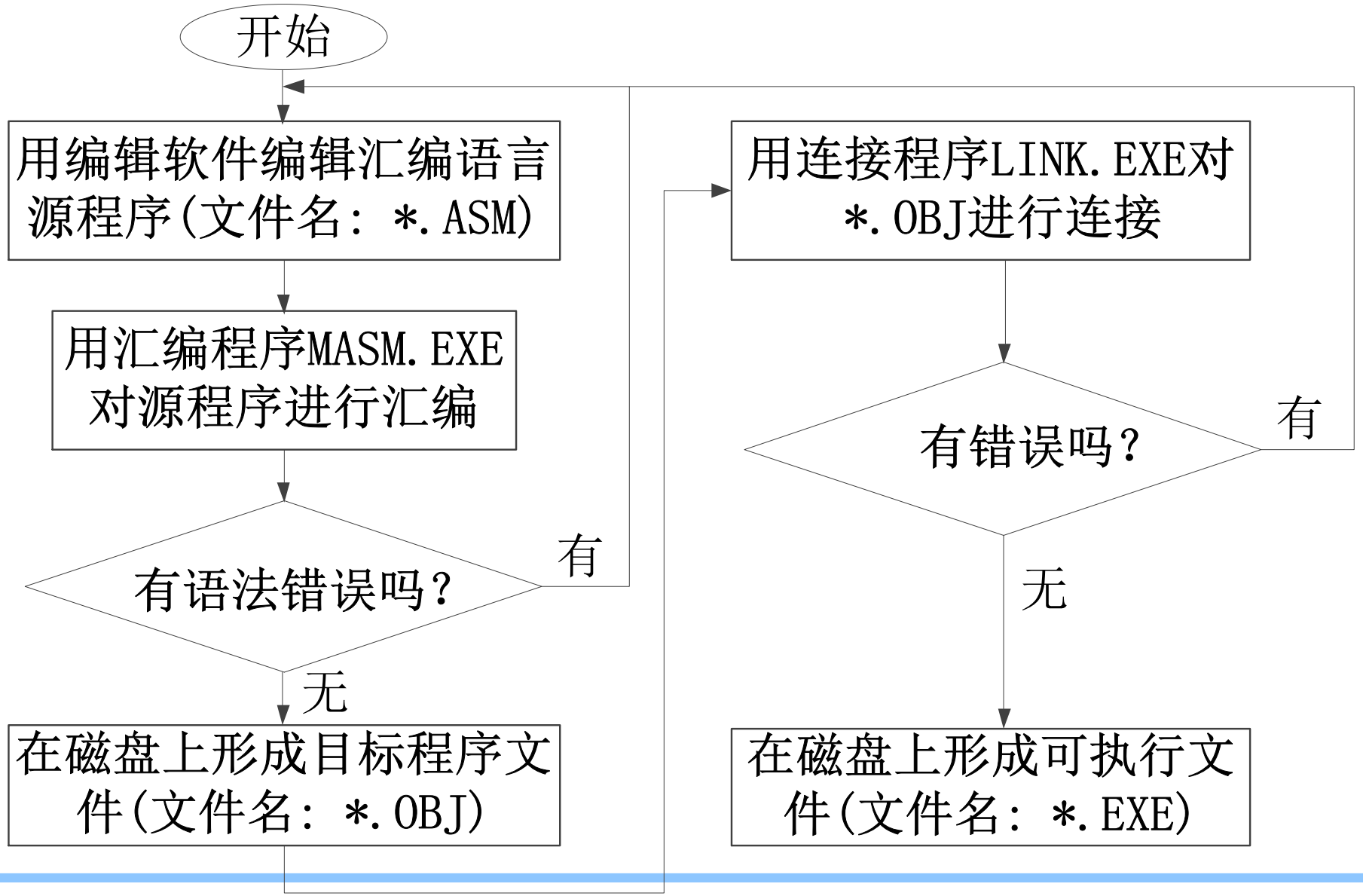
把汇编语言源程序翻译成机器语言程序的过程称为汇编。

（6）汇编程序

能把汇编语言源程序翻译成机器语言程序的系统程序（语言加工程序）。8086宏汇编程序为MASM.EXE。

（7）从汇编语言源程序到可执行程序所经过的处理过程

第3章 8086CPU指令系统



第3章 8086CPU指令系统

编译

把**汇编语言代码翻译成目标机器指令**，目标文件中所存放的是与源程序等效的机器语言代码。

连接

- **多个源程序**编译成为目标文件后，用**连接程序将它们连接到一起**；
- **只有一个源程序**，不需要调用某个库中的子程序的情况下，也**必须用连接程序对目标文件进行处理**，生成可执行文件；
- 程序中**调用某个库文件中的子程序**，需要将这个库文件和该程序生成的目标文件连接到一起。

2. 汇编语言与机器语言的联系与区别

- 汇编语言与机器语言属于低级语言，它们与高级语言有较大的区别，汇编语言中的语句与机器的型号密切相关。如Intel 8086系列CPU、Intel8031系列单片CPU等，CPU型号不同，其指令系统就不同，语句的书写格式也就不同。
- 计算机能够直接识别和执行的唯一语言是二进制语言，但人们采用符号语言或高级语言编写程序。为此，必须借助汇编程序或编译程序，把符号语言或高级语言翻译成二进制码组成的机器语言。

3. 指令及指令系统的定义

- 指令：在汇编语言源程序中以语句表示，汇编后形成一条机器语言，是要计算机执行某种操作的命令，计算机的程序是由一系列的指令组成的。
- 指令系统：一台计算机中所有机器指令的集合，是表征一台计算机性能的重要因素，其格式与功能不仅直接影响到机器的硬件结构，也直接影响到系统软件，影响到机器的适用范围。它的设计直接关系到计算机的硬件结构和用户的需要，当然，计算机硬件结构的设计也制约着指令系统。

4. 汇编语言程序中语句的种类

在汇编语言程序设计中，程序中的语句有三类：**指令语句，伪指令语句，宏指令语句。**

（1）指令语句

汇编后能产生机器语言代码，是CPU能执行的语句。

（2）伪指令语句

汇编后不能产生机器语言代码，是CPU不能执行的语句。只是告诉汇编程序（MASM.EXE）如何汇编。

（3）宏指令语句

它是8086指令系统中没有的指令，是用户自己根据宏指令定义的方法定义的一条能完成某一特定功能的新的指令。

3.1.1 汇编语言中语句的组成

汇编语言源程序（文件名.ASM）是由一条条语句组成的。语句通常由标识符，操作助记符，操作数，注释四部分组成。

其基本格式如下：

[标识符]	空格或:	**操作助记符**	空格	**操作数**	[; 注释]

◆ 操作助记符： 指出该条语句的基本操作功能，是必须有的部分。

[ ]项： 可有可无，视情况而定。

- 若是指令语句，标识符就是一个标号名，以冒号结尾；
- 若是伪指令语句，标识符就是变量名或者段名等，以空格结尾。
- 标识符的第一个字符必须是字母，不能为数字，总字符个数不能超过31个。在给标识符起名时，不能用8086指令系统中的专用符来给标识符起名称，如ADD、MOV等。

3.1.1 汇编语言中语句的组成

指令：

EXP1.　　L1:　MOV　AX，BX；BX值赋给AX

EXP2.　　L2:　MOV　AX，2000H

伪指令：

EXP1.　　VALUE1　DB　03H,04H；定义变量

EXP2.　　VALUE2　DW　1234H

EXP3.　　VALUE3　DB　3 DUP (?)

3.1.2 汇编语言中的常数与表达式

◆ 操作数

既可以是常数或表达式（即：立即数），也可以是指明操作数所在地址的一种说明。

如果操作数是常数或表达式，则有以下几种形式。

1. 数值常数

若为数值常数，则按其基数的不同，可有不同的形式。

2. 字符串常数

字符串常数是由单引号′……′括起来的一串字符或者单个字符。

3.1.2 汇编语言中的常数与表达式

3. 表达式

语句中的操作数项也可以是表达式。表达式由操作数和操作符组成。操作符有：

- 算术操作符：+、-、*、/、MOD
- 逻辑操作符：ADD、OR、XOR、NOT
- 操作符：EQ(相等)、NE(不等)、LT(less than小于)、 GT(大于)、LE(小于或等于)、GE(大于或等于)
- 属性操作符：SEG、OFFSET、TYPE、LENGTH、SIZE
- 属性修改操作符：PTR

3.1.2 汇编语言中的常数与表达式

算术操作符

- **MOV AL , 5+2*3 等效于 MOV AL , 11**
- **MOV AL , 11/2 等效于 MOV AL , 05H; 取商**
- **MOV AL , 11 MOD 2 等效于MOV AL , 01H;取余**

3.1.2 汇编语言中的常数与表达式

逻辑操作符

MOV AL , 0CCH AND 0F0H

等效于

MOV AL , 0C0H

CPU执行时完成的操作

汇编程序汇编时完成的操作

AND AL , 0CCH OR 0F0H

等效于

AND AL , 0FCH

3.1.2 汇编语言中的常数与表达式

关系操作符

- 若关系成立，则为真，取值全1；
- 若关系不成立，则为假，取值全0。

MOV AL , 04H LT 05H ;关系成立为真

等效于

MOV AL , 0FFH

3.1.2 汇编语言中的常数与表达式

属性操作符

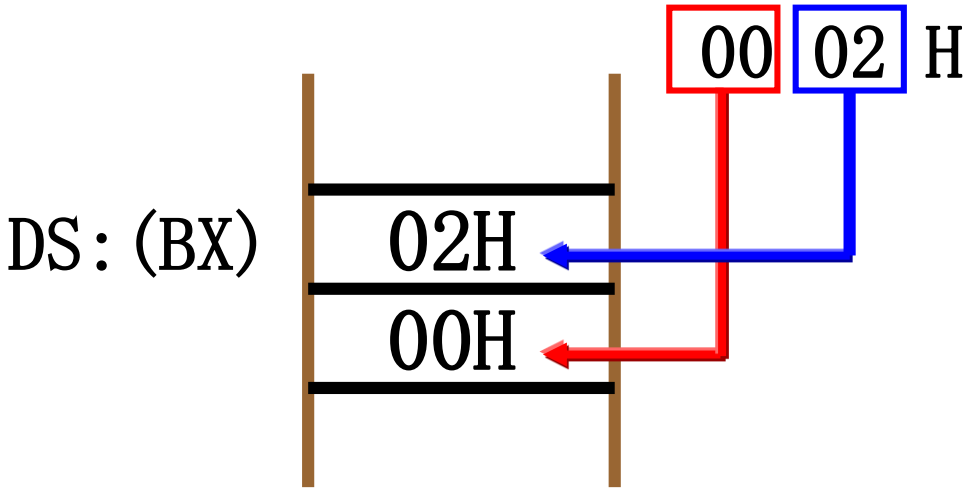
```
MOV  BX, OFFSET TABLE;   BX取变量TABLE单元的偏移地址
MOV  AX, SEG  TABLE;     AX取变量TABLE单元的段地址
MOV  DL, TYPE TABLE;     DL取变量TABLE的类型
```

均为立即数寻址

3.1.2 汇编语言中的常数与表达式

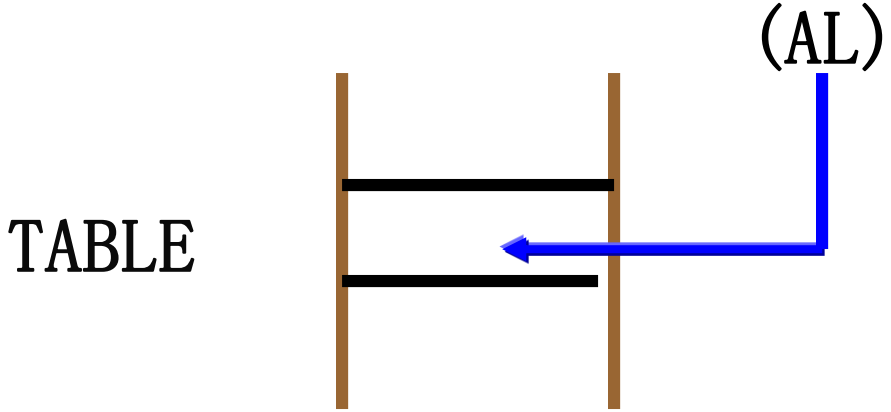
属性修改操作符

MOV WORD PTR [BX] ,02H



3.1.2 汇编语言中的常数与表达式

MOV BYTE PTR TABLE, AL



注：只是在本条语句中用**PTR**将**TABLE**变量临时修改为字节型变量，脱离了本条语句，变量**TABLE**的类型恢复为原类型。

3.1.3 标号、变量及伪指令

对于伪指令，汇编语言程序设计中，为方便记忆，**将直接访问的存储器单元的实际地址符号化**，即给要访问存储器单元起一个标识符名，该**标识符为变量名**。对于**指令则称为标号名**。

1. 标号

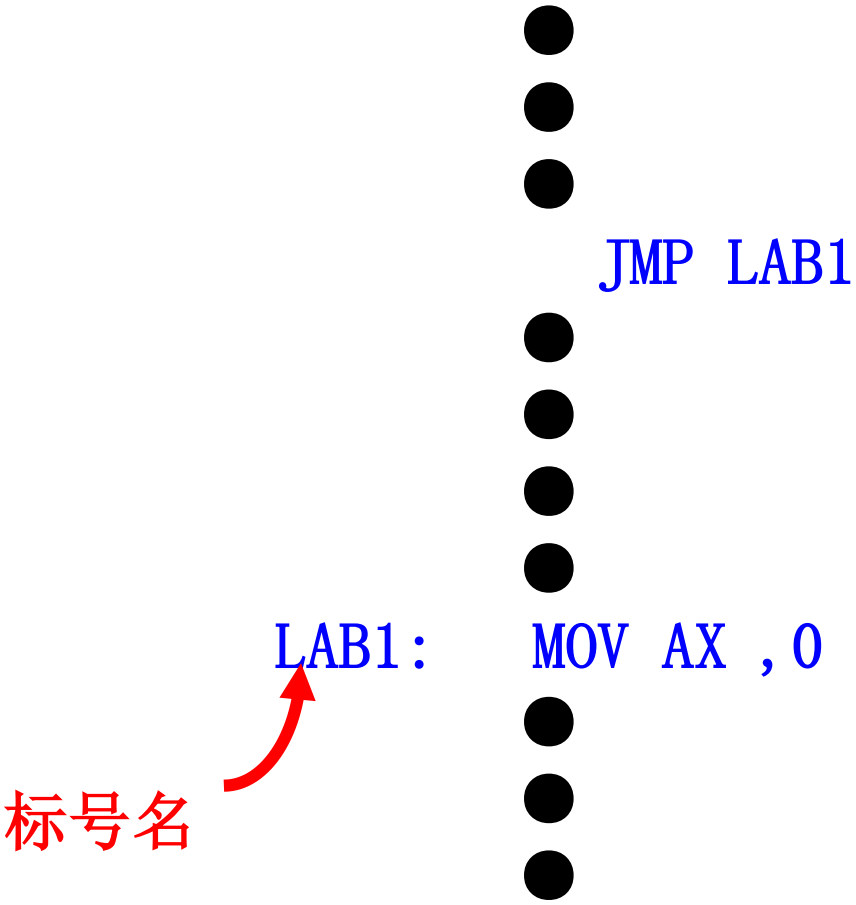
用以指示某条指令语句的位置（地址）。其定义方法就是在指令语句的操作助记符前加上标号名，以冒号结尾。它可以作为程序转移指令的操作数。

标号一旦定义，就具有了以下三个属性：

- 段地址：标号对应的指令所在段的段地址
- 段内偏移地址：标号对应指令所在的段的EA
- 类型：NEAR型 该标号与转移指令在同一代码段。
  FAR型 该标号与转移指令不在同一代码段。

3.1.3 标号、变量及伪指令

标号



3.1.3 标号、变量及伪指令

2. 变量

用以指示存放数据的存储器单元的**符号地址（地址的符号表示或地址的符号化）**。变量所指明的存储器单元的值（存储器单元的内容）在程序运行期间是可以改变的。

3. 变量定义伪指令

也称为数据定义伪指令或者存储器分配伪指令；

变量定义伪指令主要应用在数据段，是用来**给变量名所对应的存储器单元分配数据或预留空间**。

3.1.3 标号、变量及伪指令

变量定义伪指令有以下五种：

- [变量名] DB　表达式；定义字节型变量（1字节）
- [变量名] DW　表达式；定义字型变量（2字节）
- [变量名] DD　表达式；定义双字型变量（4字节）
- [变量名] DQ　表达式；定义长字型变量（8字节）
- [变量名] DT　表达式；定义一个10字节的变量

常用的变量定义伪指令有DB、DW、DD。伪指令左边的变量名可有可无，若有必须以空格结尾。

3.1.3 标号、变量及伪指令

变量定义伪指令语句中的表达式有以下几种情况：

（1）1个或多个常数或表达式，当为多个时，其间用逗号分割；

（2）带引号的字符串；

（3）一个问号（？），用来将此单元保留，存放结果；

（4）重复方式，

其格式为：重复次数　DUP　（表达式）。

3.1.3 标号、变量及伪指令

变量属性（变量定义五属性）：

- 段地址：变量所在段的段地址；
- 段内偏移地址：变量对应单元的偏移地址；
- 类型：为每个变量所占的字节数，对于DB、DW、DD、DQ、DT定义的变量**其类型分别为1，2，4，8，10**。通常又将DB、DW、DD所定义的变量称为BYTE类型，WORD类型和DWORD类型变量；
- 长度：变量定义语句中，**第一个DUP前的系数，表示变量重复的次数**，当变量定义语句中**没有出现DUP或者第一个为数据时，则其长度为1**；
- 大小：变量定义时所占用的**所有字节数**，它等于变量的长度与类型（字节数）之积 ，即SIZE=LENGTH×TYPE。

3.1.3 标号、变量及伪指令

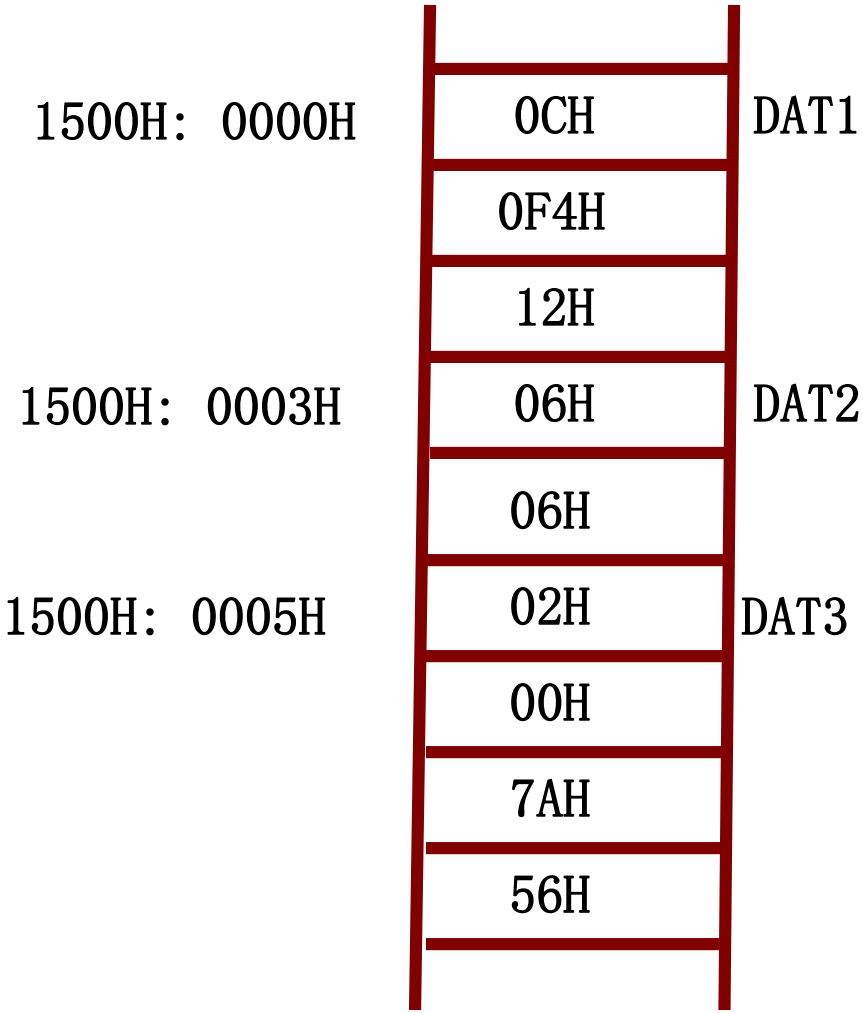
变量定义举例

假设（DS）=1500H，且在数据段0000H偏移地址开始有以下变量定义。

汇编程序对本段汇编后，各变量对应存储器单元的内容如下图。

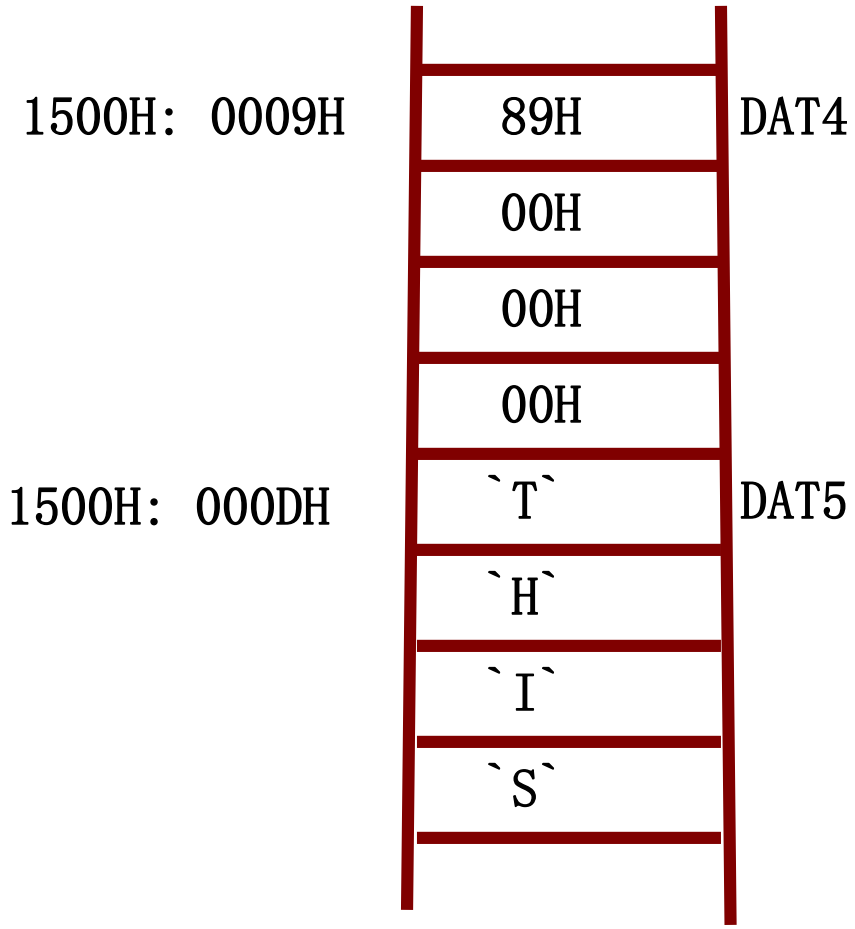
3.1.3 标号、变量及伪指令

```
DATA   SEGMENT
DAT1 DB 12，-12，12H
DAT2 DB 2*3，$+2
DAT3 DW 02H，567AH
DAT4 DD 89H
DAT5 DB 'THIS'
DAT6 DW 'AB', 'C'
DAT7 DB 3 DUP (00H)
DAT8 DW 3 DUP (?)
DAT9 DW DAT6
DAT10 DD DAT8
DAT11 DB ?
DATA  ENDS
```



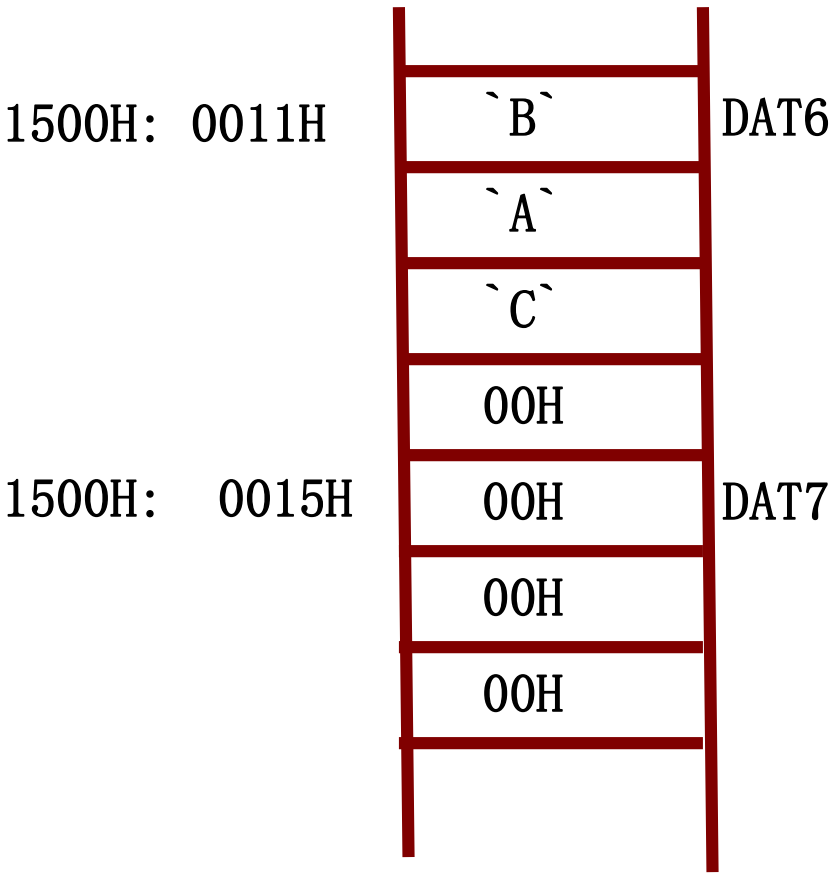
3.1.3 标号、变量及伪指令

```
DATA   SEGMENT
DAT1 DB 12，-12，12H
DAT2 DB 2*3，$+2
DAT3 DW 02H，567AH
DAT4 DD 89H
DAT5 DB 'THIS'
DAT6 DW 'AB', 'C'
DAT7 DB 3 DUP (00H)
DAT8 DW 3 DUP (?)
DAT9 DW DAT6
DAT10 DD DAT8
DAT11 DB ?
DATA  ENDS
```



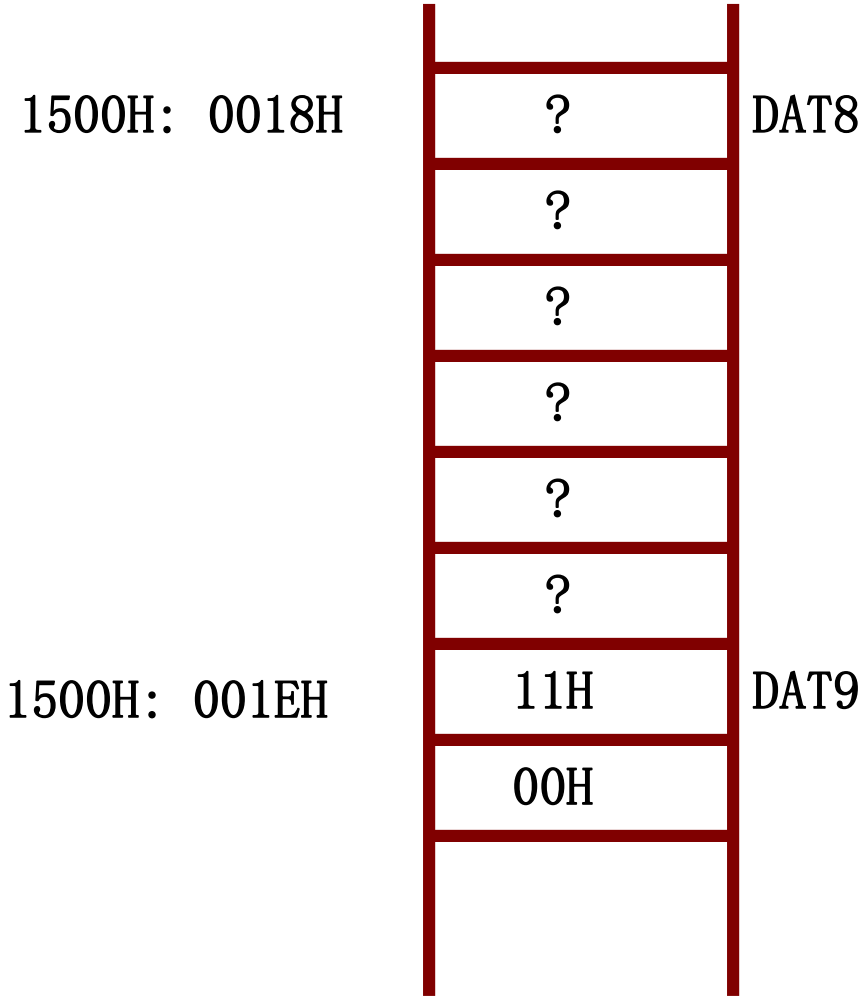
3.1.3 标号、变量及伪指令

```
DATA   SEGMENT
DAT1 DB 12，-12，12H
DAT2 DB 2*3，$+2
DAT3 DW 02H，567AH
DAT4 DD 89H
DAT5 DB 'THIS'
DAT6 DW 'AB', 'C'
DAT7 DB 3 DUP (00H)
DAT8 DW 3 DUP (?)
DAT9 DW DAT6
DAT10 DD DAT8
DAT11 DB ?
DATA  ENDS
```



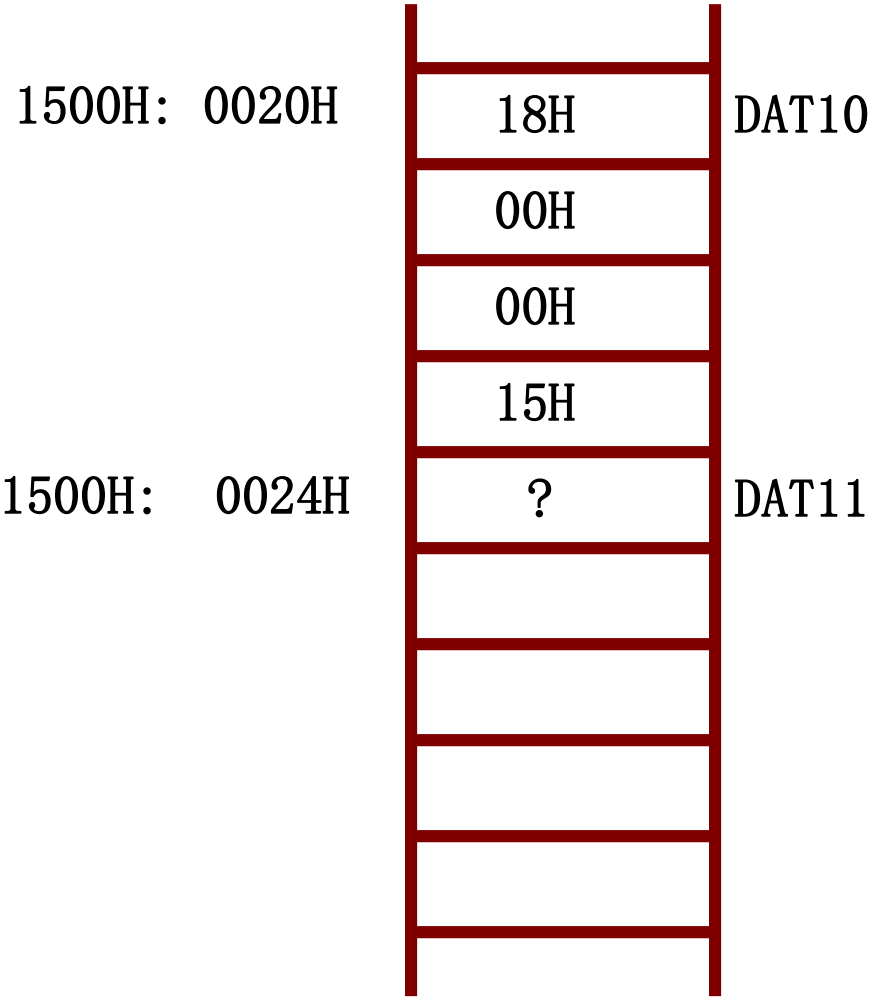
3.1.3 标号、变量及伪指令

```
DATA   SEGMENT
DAT1 DB 12，-12，12H
DAT2 DB 2*3，$+2
DAT3 DW 02H，567AH
DAT4 DD 89H
DAT5 DB 'THIS'
DAT6 DW 'AB', 'C'
DAT7 DB 3 DUP (00H)
DAT8 DW 3 DUP (?)
DAT9 DW DAT6
DAT10 DD DAT8
DAT11 DB ?
DATA  ENDS
```



3.1.3 标号、变量及伪指令

```
DATA   SEGMENT
DAT1 DB 12，-12，12H
DAT2 DB 2*3，$+2
DAT3 DW 02H，567AH
DAT4 DD 89H
DAT5 DB 'THIS'
DAT6 DW 'AB', 'C'
DAT7 DB 3 DUP (00H)
DAT8 DW 3 DUP (?)
DAT9 DW DAT6
DAT10 DD DAT8
DAT11 DB ?
DATA  ENDS
```



3.1.3 标号、变量及伪指令

伪指令DW、DD的特殊用法:

（1）<变量名1> DW <标号（或变量名2）>±常数

这里定义的<变量名1>为字型地址指针，其内容为〈<标号>±常数〉或〈<变量名2>±常数〉的段内偏移地址。例如：

```
AD1  DB  100 DUP(?)
AD2  DW  AD1                ；指向变量AD1的起始地址
AD3  DW  AD1+10
```

（2）<变量名1> DD <标号（或变量名2）>±常数

这里定义的变量名1为双字型地址指针，第一个字存放〈<标号>±常数〉或〈<变量名2>±常数〉的段内偏移地址，第二个字存放其段地址。例如：

```
AD4  DD  AD1                ；指向变量AD1的起始地址
```

3.1.3 标号、变量及伪指令

变量的属性与属性操作符

一个变量一旦定义，就具有了以下五个属性：

- 段地址（SEG）
- 段内偏移地址（OFFSET）
- 类型（TYPE）
- 长度（LENGTH）
- 大小（SIZE）

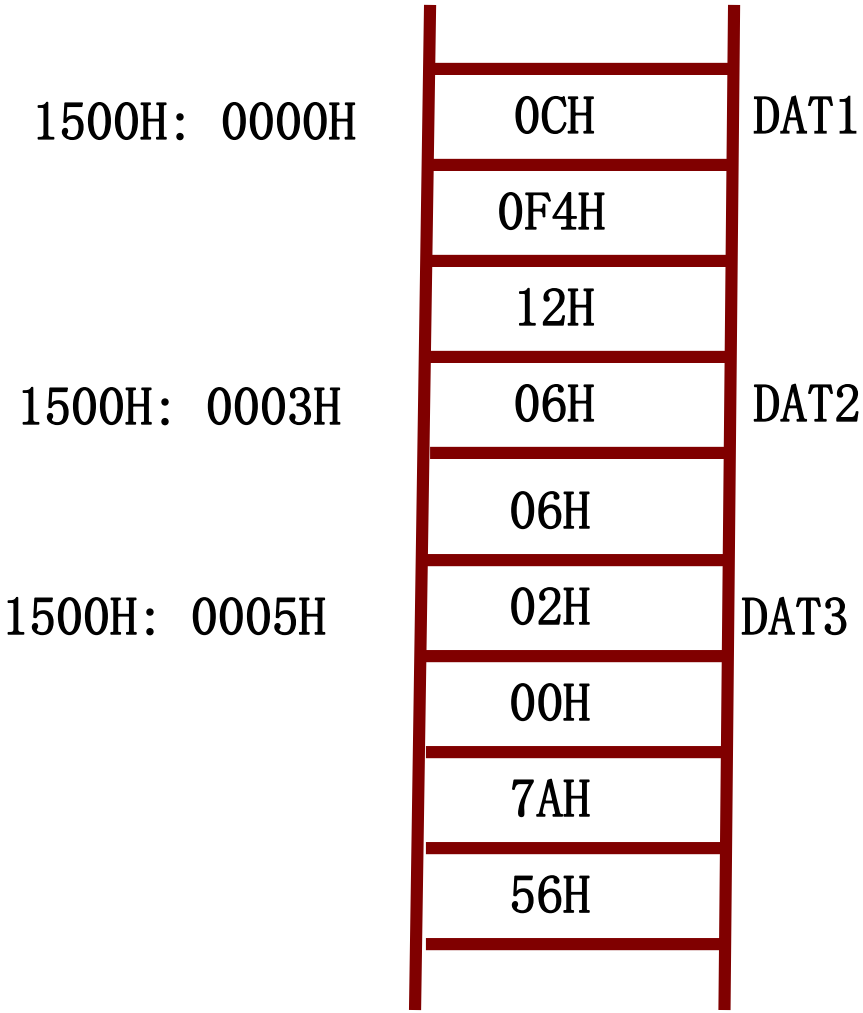
3.1.3 标号、变量及伪指令

对于前面变量定义例子中各变量的定义，则有：

```
MOV AX , SEG DAT1          ; (AX)=1500H
MOV AX , SEG DAT10         ; (AX)=1500H
MOV AX , OFFSET DAT3       ; (AX)=0005H
MOV AL , TYPE DAT3         ; (AL)=02H
```

3.1.3 标号、变量及伪指令

```
DATA   SEGMENT
DAT1 DB 12，-12，12H
DAT2 DB 2*3，$+2
DAT3 DW 02H，567AH
DAT4 DD 89H
DAT5 DB 'THIS'
DAT6 DW 'AB', 'C'
DAT7 DB 3 DUP (00H)
DAT8 DW 3 DUP (?)
DAT9 DW DAT6
DAT10 DD DAT8
DAT11 DB ?
DATA  ENDS
```



3.1.3 标号、变量及伪指令

```
MOV AX , LENGTH DAT3        ; (AX)=0001H
MOV AX , LENGTH DAT8        ; (AX)=0003H
MOV AX , SIZE DAT3          ; (AX)=0002H
MOV AX , SIZE DAT8          ; (AX)=0006H
```

以上指令中，
SRC均为立即数寻址

3.1.3 标号、变量及伪指令

```
DATA   SEGMENT
DAT1 DB 12，-12，12H
DAT2 DB 2*3，$+2
DAT3 DW 02H，567AH
DAT4 DD 89H
DAT5 DB 'THIS'
DAT6 DW 'AB', 'C'
DAT7 DB 3 DUP (00H)
DAT8 DW 3 DUP (?)
DAT9 DW DAT6
DAT10 DD DAT8
DAT11 DB ?
DATA  ENDS
```