第4章
汇编语言及其程序设计(II)

4.5 常用伪指令

1、符号定义伪指令

给一个符号重新命名；

有等值语句EQU、=、LABEL；

1) 赋值语句(EQU伪指令)

格式：符号名　　EQU 表达式

EQU将右侧表达式的值赋给左侧的符号名，经EQU定义的符号名只能定义一次，不占内存单元。

【例】　A1　EQU　30；A1代表常数30

　　A2　EQU　BX；A2代表寄存器BX

2) 等值语句(“=”伪指令)

格式：符号名　　=　　表达式

“=”伪指令与EQU具有相同的功能，区别在于使用等号“=”定义过的符号可以被重新定义，使其具有新的值，不占内存单元。

【例】　A1　=　10　　;

　　A1　=　20　　; A1被重新定义

3) 别名定义语句(LABEL伪指令)

格式：变量或标号名 LABEL 类型符

把LABEL伪指令的下一条语句中变量或标号取了一个新名字，并给这个新名字定义新的类型属性。LABEL伪指令可以定义标号的类型是NEAR还是FAR属性，定义的变量类型可以是BYTE、WORD、DWORD等类型。

2、数据伪指令语句

1) 为数据分配存储单元，并给所分配的第一个存储单元指定变量名，同时将相应的存储单元初始化，即为存储单元赋初值；

格式：[变量名] 伪指令指示符 [操作数1，操作数2，…，操作数n] [；注释]

变量名表示所分配的存储单元中第一个单元的地址；伪指令指示符有：

- DB定义字节，操作数占用一个字节空间；
- DW定义字，操作数占用一个字空间；
- DD定义双字，操作数占用两个字空间；
- DQ定义四字变量，操作数占用四个字空间
- DT定义一个十字节变量，占用10个字节空间

【例】在给定的数据段中，分析数据定义伪指令的使用和存储单元的初始化

```
B1      DB      10H，30H
                                ；存入2个字节数据
B2      DB      2×3+5
                                ；存入表达式
S1      DB      ‘GOOD!‘
                ；按字节存入5个字符的ASCII码
S2      DW      ‘AB‘
            ；按字数据存入2个字符的ASCII码
W1      DW      1000H，2030H
                                ；存入2个字数据
W2      DD      12345678H
                                ；存入双字
```

2) 重复操作符DUP

当同样的操作数重复多次时，可采用重复操作符DUP来表示。

格式：n DUP(初值[，初值，...])

圆括号中为重复的内容，n为重复次数。如果用n DUP(？)作为数据定义伪指令的唯一操作数，则汇编程序产生一个相应的数据区，但不赋予任何初始值。

重复操作符可以嵌套。

【例】

```
BUF1    DB      ?
                ；分配字节变量存储单元，不赋初值
BUF2    DB  8   DUP(0)
        ；给字节变量BUF2分配8个存储单元，并赋初值为0
BUF3    DW  5   DUP(？)
；给字变量BUF3分配5个字单元，即10个存储单元，不预
                                赋初值
BUF4    DW  10  DUP(0，1，？)
  ；给字变量BUF4分配初始数据0、1、？且重复次数为10
                        个存储空间，共占60字节
BUF5    DB      50  DUP(2，2，DUP(4)，6)
  ；给字节变量BUF5定义为一个数据区，其中包含重复50
                次的内容：2、4、4、6，共200字节
```

3、段定义伪指令

主要有2条，SEGMENT/ENDS和ASSUME。

1) SEGMENT/ENDS　　段定义伪指令

用来定义各种类型的逻辑段。格式：

```
段名    SEGMENT
……                    ；段内所有语句
段名    ENDS
```

2) ASSUME　伪指令

段分配语句用来完成段分配功能。格式：

ASSUME 段寄存器名：段名　　[，段寄存器名：段名，...]

ASSUME语句应安排在代码段开始，指出段名与段寄存器名之间的关系，当前定义过的逻辑段分别被设为代码段、数据段、堆栈段或附加段中的一个，段名是用段定义伪指令定义过的名字，段寄存器名可以是CS、DS、ES、SS中的一个。

KA1	42H	0200：0000H
	41H	0001H
	42H	0002H
	41H	0003H
	…	
KA2	56H	0028H
	12H	0029H
	00H	002AH
	00H	002BH
KA3	41H	002CH
	42H	002DH
	43H	002EH
	44H	002FH
	45H	0030H
	46H	0031H
	47H	0032H
	48H	0033H
KA4	00H	0034H
	…	
	00H	003DH

【例1】已知数据段DATA从存储器实际地址02000H开始

```
DATA    SEGMENT
KA1 DW      20  DUP(4142H)
KA2 DD      1256H
KA3 DB      'ABCDEFGH'
KA4 DB      10  DUP(0)
DATA    ENDS
```

MOV指令单独执行后，各寄存器的内容是?

KA1	42H	0200：0000H
	41H	0001H
	42H	0002H
	41H	0003H
	…	
KA2	56H	0028H
	12H	0029H
	00H	002AH
	00H	002BH
KA3	41H	002CH
	42H	002DH
	43H	002EH
	44H	002FH
	45H	0030H
	46H	0031H
	47H	0032H
	48H	0033H
KA4	00H	0034H
	…	
	00H	003DH

(1)MOV　AL，TYPE　KA1

(2)MOV　AH，TYPE　KA2

(3)MOV　BL，TYPE　KA3

TYPE类型属性运算符——该运算返回变量或标号的类型值。标号NEAR、FAR的类型值TYPE分别为-1、-2

变量类型属性TYPE用来回送变量中每一个数据项的长度(以字节为单位)，变量类型DB、DW、DD、DQ和TD的TYPE返回值分别为1、2、4、8和10。

(1)AL = 2，(2)AH = 4，(3)BL = 1

(4)MOV　AX，KA1

直接寻址方式，取出KA1字单元中的内容4142H送AX，AX = 4142H

KA1	42H	0200：0000H
	41H	0001H
	42H	0002H
	41H	0003H
	…	
KA2	56H	0028H
	12H	0029H
	00H	002AH
	00H	002BH
KA3	41H	002CH
	42H	002DH
	43H	002EH
	44H	002FH
	45H	0030H
	46H	0031H
	47H	0032H
	48H	0033H
KA4	00H	0034H
	…	
	00H	003DH

(5)MOV　CL，LENGTH　KA1

(6)MOV　CH，LENGTH　KA2

LENGTH运算符——用来回送分配给该变量的单元数。当变量是用重复数据操作符DUP定义的，则返回DUP前面的数值(即重复次数)；如果没有DUP说明，则返回值总是“1”。

(5)CL = 20，(6)CH = 1

(7)MOV　DX，SIZE　KA1

(8)MOV　BP，SIZE　KA2

SIZE运算符——返回变量所占的总字节数，即

$$SIZE = TYPE \times LENGTH$$

(7)DX = 2×20 = 40

(8)BP = 4×1 = 4

KA1	42H	0200：0000H
	41H	0001H
	42H	0002H
	41H	0003H
	…	
KA2	56H	0028H
	12H	0029H
	00H	002AH
	00H	002BH
KA3	41H	002CH
	42H	002DH
	43H	002EH
	44H	002FH
	45H	0030H
	46H	0031H
	47H	0032H
	48H	0033H
KA4	00H	0034H
	…	
	00H	003DH

(9)MOV SI，SEG KA1

(10)MOV DI，SEG KA4

SEG取段地址运算符——该运算符返回变量或标号所在段的段地址。

(9)SI = 0200H，(10)DI = 0200H

(11)MOV DX，OFFSET KA1

(12)MOV AX，OFFSET KA4

OFFSET取段内偏移量符——该运算返回变量或标号所在段的段内偏移量。

(11)DX = 0，(12)AX = 0034H

4、过程定义伪指令

1) 过程定义伪指令PROC/ENDP

格式:

过程名　PROC　　[NEAR]/FAR

　　……

　　RET

　　……

过程名　ENDP

2) 过程调用及返回

格式:

CALL　　过程名

5、地址定位伪指令ORG

指明数据或程序存放的起始地址的偏移量，即从表达式提供的偏移地址开始存放。

格式：ORG　　数值表达式

6、汇编指针计数器$

汇编指针用符号“$”表示，在程序中，“$”出现在表达式里，表示汇编到该伪指令后分配内存单元的偏移地址。

7、模块定义与连接伪指令END

1) NAME伪指令：用于给源程序汇编后得到的目标程序指定一个模块名，连接时需要使用这个目标程序的模块名
2) END伪指令：源程序到此结束，指示汇编程序到此结束，对于END后面的语句可以不理会；
3) PUBLIC伪指令：说明本模块中的某些符号是公共的；
4) EXTRN伪指令：说明本模块中所用的某些符号是外部的，在定义这些符号的模块中还必须用PUBLIC伪指令加以说明；

【例2】已知数据段定义如下

```
DATA        SEGMENT
            ORG 200H
X1          DW  5，$+8，6，7
X2          EQU $-X1
X3          DB  7，8，X2，9
DATA        ENDS
```

问执行指令　　MOV　　AX，X1+2

和　　MOV　　BL，X3+1

后，AX =（　　），BL =（　　）?

地址定位伪指令ORG—指明数据或程序存放的起始地址的偏移量，即从表达式提供的偏移地址开始存放。本题将偏移量定为0200H。

汇编指针计数器$—在程序中“$”出现在表达式里，表示汇编到该伪指令后分配内存单元的偏移地址。

变量	值	逻辑地址
X1	05H	DS：0200H
	00H	0201H
	0AH	0202H
	02H	0203H
	06H	0204H
	00H	0205H
	07H	0206H
	00H	0207H
X3	07H	0208H
	08H	0209H
	08H	020AH
	09H	020BH

```
DATA    SEGMENT
        ORG  200H
X1      DW   5，$+8，6，7
X2      EQU  $-X1
X3      DB   7，8，X2，9
DATA    ENDS
```

$第1出现，分配内存单元的偏移地址为0202H，值：0202H+8H = 020AH

赋值语句EQU——将右侧表达式的值赋给左侧的符号名；

$第2出现，分配内存单元的偏移地址为0208H，X1的偏移地址为0200H，因此：

X2 = 0208H-0200H = 8H

执行指令MOV　AX，X1+2

AX = (DS×16+0200H+0002H) = 020A

执行指令MOV　BL，X3+1

BL = (DS×16+0208H+0001H) = 08H

变量	值	逻辑地址
K1	41H	DS：0000H
	42H	0001H
	43H	0002H
	44H	0003H
K2	45H	0004H
	23H	0005H

K1的偏移量为0，K2的偏移量为4，ORG将偏移量定为200H，$出现，分配内存单元的偏移地址为200H，K3 = 200H-0 = 200H

1) BX = K2的偏移地址0004H
2) AX = 0200H
3) 是偏移量K1+3，K1+4的存储单元，CX = 4544H

【例3】数据段定义如下

```
DATA    SEGMENT
K1      DB   'ABCD'
K2      DW   2345H
        ORG  200H
K3      EQU  $-K1
DATA    ENDS
```

问各条MOV指令单独执行后，各寄存器的内容是多少？

```
(1)MOV   BX，OFFSET   K2
(2)MOV   AX，K3
(3)MOV   CX，WORD     PTR  K1+3
```

变量	值	段地址：偏移地址
K1	41H	2000：0200H
	42H	0201H
	43H	0202H
	44H	0203H
	45H	0204H
K2	24H	0205H
	00H	0206H
K4	00H	0207H
	02H	0208H
K5	00H	0209H
	02H	020AH
	00H	020BH
	20H	020CH
K6	13H	0213H
	02H	0214H

【例4】设DATA值为2000H，写出K1，K2，K3，K4，K5，K6的偏移地址或常数值。

```
DATA    SEGMENT
        ORG  200H
K1      DB   'ABCDE'
K2      DW   24H
K3      EQU  K2-K1
K4      DW   K1
K5      DD   K1
K6      DW   $+6
DATA ENDS
```

K3 = 5

$+6 = 020DH+06H = 0213H

4.6 宏指令与高级汇编技术

1、宏汇编及处理程序

“宏”是指源程序中一段具有独立功能的代码，宏指令代表一段源程序，必须先定义后调用。

语句格式:

```
宏名 MACRO    [形式参数表]          ；宏定义
...                                 ；宏体
      ENDM                          ；宏定义结束
```

宏汇编时，宏指令被自动地展开成为相应的机器码而插入源程序中的宏调用处，宏调用指令在汇编过程中对宏定义体作宏展开操作，宏定义体取代源程序中的红指令名，并用实参取代宏定义中的形参。

【例】为实现ASCII码与BCD码之间相互转换，可把寄存器AL中的内容左移或右移4位。定义一条实现将AL中的内容左移4为的宏指令并进行调用。

源程序：

```
SHIFT MACRO                ；宏指令名为SHIFT
          MOV  CL，4       ；计数器CL赋初值4
          SAL  AL，CL      ；对AL中内容算术左移4位
ENDM                       ；宏定义结束
          MOV  AL，20H     ；对AL赋初值
          ADD  AL，25H     ；进行加法处理
          SHIFT            ；宏调用
          MOV  [BX]，AL    ；结果保存在内存单元
```

程序中凡要使AL中的内容左移4位，就可用宏指令SHIFT来代替。

❖ 定义带参数的宏定义和宏调用，对给定寄存器内容进行移位操作，满足不同移位次数。

若取1个参数，设移位次数为CN

```
SHIFT MACRO     CN
          MOV   CL，CN
          SAL   AL，CL
ENDM
```

宏调用时提供具体的实参，如移位4次

```
          SHIFT 4
```

若取2个参数，设移位次数为CN

```
SHIFT MACRO     CN，R
          MOV   CL，CN
          SAL   R，CL
ENDM
```

宏调用时提供具体的移位次数和寄存器名，可对任一个寄存器实现指定的左移次数

```
          SHIFT 4，AL       ；对AL寄存器移位4次
          SHIFT 8，AX       ；对AX寄存器移位8次
```

2、宏和子程序比较

宏和子程序都可用来简化源程序，并可多次对它们进行调用，从而使程序结构间接清晰。对于需重复使用的程序模块，可用子程序也可用宏来实现。

主要区别：

①宏操作可直接传递和接收参数，不需通过堆栈等其它媒介来进行，编程比较容易；

②宏调用只能简化源程序的书写，缩短源程序长度，并没有缩短目标代码的长度，汇编程序处理宏指令时把宏体插到宏调用处，目标程序占用内存空间并不因宏操作而减少；

③引入宏操作并不会在执行目标代码时增加额外的时间开销；相反，子程序调用由于需要保护和恢复现场及断点，会延长目标程序的执行时间；

3、重复汇编

重复汇编可以出现在宏定义中，也可出现在源程序的任何位置上。有3种形式:

①定重复伪指令REPT/ENDM。重复汇编的次数用表达式的值来表示。

②不定重复伪指令IRP/ENDM。重复汇编的次数由参数表中的个数决定。

③不定重复字符伪指令IRPC/ENDM。重复汇编的次数等于字符串中字符的个数。

【例】使用不同的重复伪指令定义10个数据，将数据0、1、2、…、9分配给10个连续的字节单元

方法一：定重复伪指令REPT/ENDM

```
COUNT = 0
REPT  10
      DB    COUNT
      COUNT = COUNT + 1
ENDM
```

方法二：不定重复伪指令IRP/ENDM

```
IRP   X，<0，1，2，3，4，5，6，7，8，9>
      DB    X
ENDM
```

方法三：不定重复字符伪指令IRPC/ENDM

```
IRPC  X，0123456789
      DB    X
ENDM
```

4、条件汇编

是指汇编程序根据某条件对部分源程序有选择地进行汇编。条件汇编语句是一种说明性语句，其功能由汇编系统实现。通常在宏定义中使用。

条件伪操作的一般格式:

```
IF          <表达式>
            [语句序列1]
ELSE
            [语句序列2]
ENDIF
```

有5组条件汇编语句：

①IF和IFE：是否为0条件语句

②IF1和IF2：是否为1条件语句

③IFDEF和IFNDEFE：是否有定义条件语句

④IFB和IFNB：是否为空条件语句

⑤IFIDN和IFDEF：字符串比较条件语句

【例】将键盘输入单个字符及屏幕显示输出单个字符的DOS功能调用放在一个宏定义中，通过判断参数为0还是非0来选择是执行输入还是输出字符。所编制的程序中含有条件汇编的语句。

源程序

```
INOUT    MACRO    X      ; 定义宏指令名为INOUT
         IF       X      ; 条件判断
         MOV AH，02H; 条件成立，输入单个字符
         INT   21H
         ELSE            ; 条件不成立，输入单个字符
         MOV AH，01H
         INT   21H
         ENDIF           ; 条件汇编结束
ENDM
```

当宏调用为INOUT　0时，...

当宏调用为INOUT　1时，...