第八章 高级宏汇编语言

§8.1 结构

结构就是将逻辑上有一定关系的一组数据，以某种方式组合在一起所形成的数据形式。

将学生的学籍档案以结构的数据形式来表示。每个学生的各种数据，如姓名、学号、年龄和各科成绩等等，组成了一个结构型数据。

一、结构的定义

结构的定义使用结构伪指令，其格式为：

```
结构名      STRUC
 < 数据定义语句序列 >
结构名      ENDS
```

在一个源程序中结构名必须是唯一的。数据定义语句序列是用DB、DW或DD定义的变量，作为结构中的各字段，所定义的各变量名称为结构字段名。

```
例如：  STUDENT     STRUC
        CLASS       DB  '27063010'
        NUM         DB  ?
        NAME        DB  15 DUP（？）
        SCORE       DB   ?
        STUDENT     ENDS
```

注意：使用伪指令STRUC/ENDS定义的结构不产生目标代码，而定义的各个字段也不分配存储单元。

二、结构的预置与存储分配

结构预置：用定义的结构设置结构变量，分配存储空间。

结构变量预置语句的格式为：

结构变量名　　结构名　　<字段值表 >

其中：

结构名使用**STRUC/ENDS**定义的结构。

结构变量名是结构副本的标识符，与其它变量一样，它也有三个属性：段、偏移量和类型属性。类型属性表示结构的总字节数。

字段值表是为结构变量中各字段的值进行预置初值，必须用一对尖括号括起来。

例如：

```
DATA  SEGMENT
STU1    STUDENT    <    >    ;不改变结构定义时的初值定义
STU2    STUDENT    <‘25060020’>;只改变第一个字段的值
STU3    STUDENT    <,10,‘LI MING’,90>;第一个字段不变,其余重置
STU4   STUDENT    10H  DUP(<‘27060010’>)
                ;同时预置10H个副本,每个副本只修改第一个字段初值。
DATA   ENDS
```

三、对结构变量及其字段的操作

结构变量的使用与一般变量一样，可以作为一条指令的操作数。

TYPE运算符作用结构变量，其返回值为该结构变量的总字节数。

例如：

```
MOV  AX，TYPE  STU1；AX<=25
MOV  BH，LENGTH  STU2；BH<=1
MOV  BL，LENGTH  STU4；BL<=10H
MOV  CX，SIZE  STU3  ；CX<=25
MOV  DX，SIZE  STU4  ；DX<=16*25
```

访问结构变量中的字段要使用结构字段运算符“.”

其使用格式为：　结构变量名. 结构字段名

结构变量的字段的使用与一般变量的使用完全相同。

例如：

```
MOV   SI , OFFSET  STU1.CLASS    ;  SI<=0
MOV   DI , OFFSET  STU2.CLASS    ; DI<=25
MOV   AX, LENGTH STU1. NAME   ; AX<=15
MOV   BX, OFFSET  STU2
MOV   [BX].SCORE, 80   ;将STU2的SCORE字段赋值80
```

四、程序举例

例1 现有结构定义和预置如下：

```
BLOCK STRUC
FB1   DB ?
FB2   DB ?
FB3   DW 10H DUP(?)
BLOCK ENDS
DATA SEGMENT
STRU_VAR BLOCK 20 DUP(< >)
DATA ENDS
```

要求在20个结构变量的FB1字段中依次存入字母A、B、C、D.....，在FB2字段中依次存入字母Z、Y、X、W、V、.....。主要程序段编制如下：

```
        MOV DI,OFFSET STRU_VAR   ;取结构变量首址
        MOV AL,'A'         ;取初始化字段内容
        MOV AH,'Z'
        MOV CX,LENGTH STRU_VAR    ;取结构变量个数
LOP:MOV [DI].FB1, AL     ;依次向字段送字母
       MOV [DI].FB2, AH
       INC AL                    ;修改字段内容
       DEC AH
       ADD DI,TYPE STRU_VAR ;修改指针
       LOOP  LOP
```

例2 设学生学籍结构为SC1～SC7是7个连续字段，分别记载每个学生的7门课程的成绩；AVERAGE字段是该学生的平均成绩。在数据段预置了30位学生的结构副本。另有程序已完成30位学生的成绩录入。要求编制一子程序计算30位学生7门课程的平均成绩并送入相应的AVERAGE字段中。

结构定义和数据段中结构预置如下：

```
STUD_SCOR STRUC
:       :
SC1             DB  0
SC2             DB  0
SC3             DB  0
SC4             DB  0
SC5             DB  0
SC6             DB  0
SC7             DB  0
AVERAGE    DB 0
STUD_SCOR ENDS
:
CLA_SCOR  STUD_SCOR 30 DUP(<>)
```

```
SCORE  PROC
       LEA BX,CLA_SCOR;取第一个结构副本首址送BX
       MOV CX,30      ;计算平均成绩的总人数
LOP1:  XOR AX,AX
       PUSH CX
       LEA SI,[BX].SC1;SI<=一位学生SC1字段偏移量
       MOV CX,07H
LOP2:  ADD AL,[SI]   ;计算一位学生的总成绩
       ADC AH,0 ;由于[SI]是字节单元,不直接使用AX
       INC SI
       LOOP  LOP2
       MOV CL,07H
       DIV CL      ;计算平均成绩
       MOV [BX].AVERAGE,AL ;存放平均成绩
       ADD BX,TYPE CLA_SCOR;指向下一个结构副本
       POP CX
       LOOP LOP1
       RET
SCORE  ENDP
```

§8.2 记录

记录与结构相似，也是一组数据定义的组合。所不同的是，结构是以字节为基本单位构成字段，而记录是以二进制数位（BIT）为基本单位构成字段。

一、记录的定义

记录名 RECORD 字段名：宽度[=表达式]，字段名：宽度[=表达式].....

记录与结构在定义格式上的区别：记录的定义只是一个语句，而结构的定义要使用多个语句。

宽度是定义该字段所需要的二进制位数。表达式是赋予字段的初值，其值不能超过宽度所能表示的正整数，为可选项。

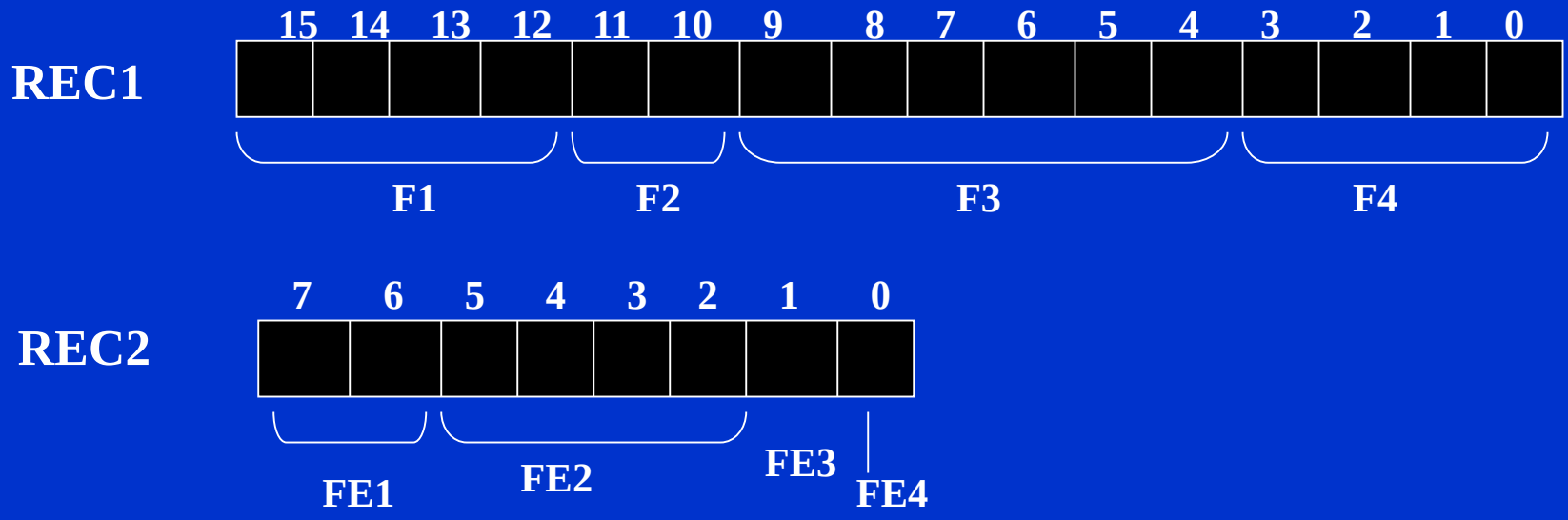
一个记录所有字段宽度之和应≤16。当字段宽度之和>8时，汇编程序自动将记录定义为字（16位），否则定义为字节（8位）。

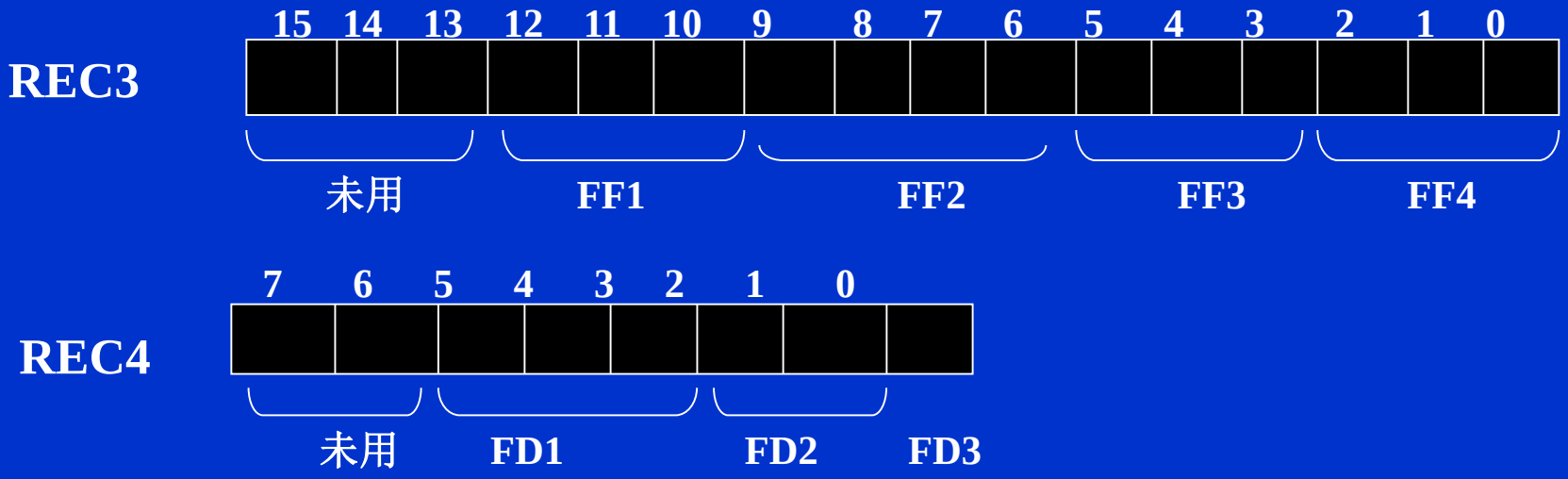
根据字段定义的顺序，先定义的在高位，最后定义的字段在最低位。如果各字段之和<8或<16，则未定义的高位以“0”填充。

例如：

```
REC1  RECORD   F1:4,F2:2,F3:6,F4:4
REC2  RECORD   FE1:2,FE2:4,FE3:1,FE4:1
REC3  RECORD   FF1:3,FF2:4,FF3:3,FF4:3
REC4  RECORD   FD1:3=100B,FD2:2=2,FD3:1=1
```

上述定义的各记录的字段分配如下图所示。





记录定义时不分配存储单元，只有当预置了记录时，才分配存储空间。

二、记录的预置与存储分配

记录变量预置语句的格式：记录变量名　记录名　〈字段值表〉

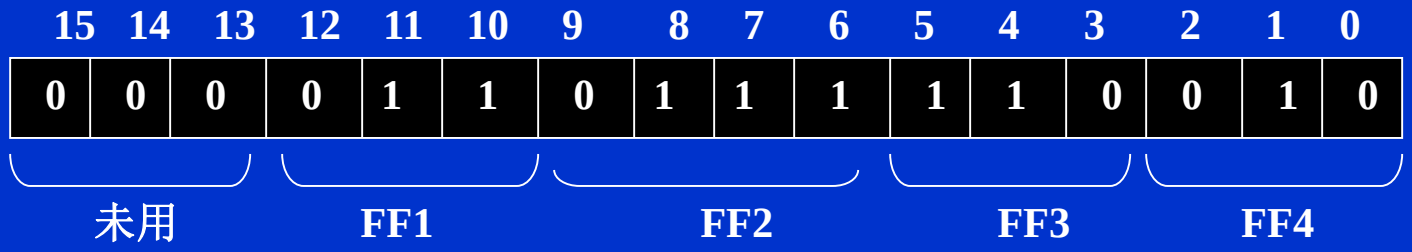
其中记录变量名是可选项，如果选用，它就是记录副本的标识符。

在字段值表中未指定初值的字段，用逗号表示，它将使用定义的初值，若定义时未指定值，则用**0**填入。

例如：

```
DA1  REC1   <0AH,3,25H,6>
DA2  REC2   <1,8,0,1>
DA3  REC3   5 DUP(<3,7,6,2>)
DA4  REC4   <3,3,0>
DA5  REC4   <,3>
DA6  REC4   10H DUP(< >)
```

上述各记录预置后，按字节或字分配存储单元。例如对记录变量**DA3**分配存储单元如下：



三、记录运算符

在宏汇编中有几个运算符是专门用于对记录进行操作。

1、记录宽度运算符（WIDTH）

WIDTH运算作用于记录名或记录字段名。其运算结果是返回该记录或记录字段的宽度（二进制数的位数）。

例如：

```
NF1  EQU  WIDTH  REC1  ;NF1=10H
NF2  EQU  WIDTH  F3    ;NF2=06H
NF3  EQU  WIDTH  F4    ;NF3=04H
     MOV  AH,WIDTH REC2;(AH)=08H
     MOV  AL,WIDTH FE1 ;(AL)=02H
     MOV  BH,WIDTH REC3;(BH)=0DH
     MOV  BL,WIDTH FF2 ;(BL)=04H
```

2. 移位值运算

在语句中如果直接将记录字段名作为一个操作数引用,则表示取该字段的最低位移到所在记录的最低位所需的移位次数。

例如：

```
MOV  BL，F1   ；  （BL）=0CH
MOV  BH，F2   ；  （BH）=0AH
MOV  CL，F3   ；  （CL）=04H
MOV  CH，F4   ；  （CH）=00H
```

3、记录屏蔽运算符

在记录字段名前加上MASK运算符，将返回该记录字段在记录中的屏蔽码。所谓屏蔽码是指该字段的各位为1其余各字段全为0，所构成的编码。

例如：

```
MOV  AX, MASK  F1     ;(AX)=0F000H
MOV  BL, MASK  FE2    ;(BL)=3CH
MOV  CX, MASK  FF3    ;(CX)=38H
MOV  BH, MASK  FD1    ;(BH)=38H
```

四、对记录及其字段的操作

1、对记录变量的操作

对预置的记录变量，就可以象普通变量一样进行存取。

例如：

```
MOV  AX，DA1 ；取出DA1记录变量的值0AE56H送AX
MOV  BL，DA2 ；BL <= 61H
MOV  DA3+4，CX ；将CX内容送DA3的第三个记录变量
```

2、对记录操作数的操作

在程序中，可以直接将已经定义的记录名作操作数使用，它被作为一个常数使用，即寻址方式为立即数寻址。

使用时，记录名后必须有一对尖括号。如果括号中无内容，表示使用记录定义时的初值。也可以为各字段重新指定值。

例如：

```
MOV  AX,REC1<0BH,2,3AH,7>;对记录REC1各字段赋值后,其值0BBA7H送AX
MOV  BL,REC4<,0,0>  ;将记录REC4的FD2和FD3字段清零后，其值20H送BL
MOV  BH,REC4< >    ;将记录REC4的原来值送BH,注意未定义的位用0填充
MOV  CL,REC4<,0,0>+REC4< >;将20H+25H的值45H送CL
```

3、对记录字段的操作

由于一个记录字段是一个字节或一个字中的某几位，在处理时，需要将记录变量作为一个整体进行操作。

例如下面的程序段是将记录变量DA1的F2字段取出，并将其移位到最右边。

```
MOV  AX,DA1          ;取记录变量
AND  AX,MASK F2 ;用屏蔽码分离出F2字段
MOV  CL,F2             ;取F2的移位值送CL
SHR  AX,CL
```

如果要修改记录变量的字段,可先取出,修改后再存回记录变量中。

§8.3 宏指令

在汇编源程序设计中，如果要多次重复使用某一个程序段，这些程序段虽然出现位置不同，但功能完全相同，或者只是修改某些操作数字段。这时可使用宏指令来实现。

使用宏指令可以使源程序更加清晰，易于阅读，简化重复程序的编写。

一. 宏功能的使用过程

宏功能的使用过程包括：宏定义，宏调用和宏展开。

1. 宏定义

宏定义使用一对伪指令**MACRO**和**ENDM**。宏定义格式有两种。

（1）不带参数的宏定义

```
宏名    MACRO
          :
          :    宏体
          :
        ENDM
```

（2）带参数的宏定义

```
宏名   MACRO   形参1，形参2，.....
          :
          :    宏体
          :
        ENDM
```

注意：宏定义本身不生成任何目标代码。

2. 宏调用

宏调用就是在源程序的任意位置直接引用已经定义的宏名。所构成的语句称为宏指令语句。

宏调用分为无参数调用和带参数调用，其格式分别为：

（1）无参数宏调用：宏名

（2）带参数宏调用：宏名　实参1，实参2，.....

例如：

```
INPUT    MACRO                    ；定义宏INPUT
         MOV  AH,01H
         INT  21H
         AND  AL,0FH
         ENDM
EXCHANGE  MACRO  BY1,BY2  ；定义宏EXCHANGE
         PUSH  AX
         MOV  AL,BY1
         XCHG AL,BY2
         MOV  BY1,AL
         POP  AX
         ENDM
         :
         INPUT                    ；调用宏INPUT
         :
         :
         EXCHANGE  DA_BY1,DA_BY2 ；调用宏EXCHANGE
         :
```

带参数宏调用时，实参与形参的排列顺序应一致。如果实参的个数比形参多，则多余的实参将被略去。如果实参的个数比形参少，则未指定的形参将用“空白串”替代。

3. 宏展开

宏展开是指汇编程序在汇编源程序过程中，当扫描到宏指令语句时，将用宏定义中的宏体的程序段目标代码替代宏指令语句。对于带参数的宏调用，将同时用相应的实参替代宏体中对应的形参。

二. 连接符&和带空格或逗号的实参

在宏定义时，可以将形参作为一个字符串中的一部分，这时需要使用连接符&。

在宏调用时，可以在实参中包含空格和逗号字符，这时需要用“<>”将实参括起来。

例如：

```
SHIFT   MACRO  VAR,REG,SHF,NUM,DEST
        MOV REG,VAR
        MOV CL,NUM
        S&SHF REG,CL   ;  SHF为形参
        MOV DEST,REG
        ENDM
        :
    SHIFT  DA_WORD,AX,AR,CONT+1,<WORD PTR DEST1+2>
```

上述宏展开后为：

```
MOV  AX,DA_WORD
MOV  CL,CONT+1
SAR  AX,CL
MOV  WORD PTR DEST1+2,AX
```

§8.4 重复汇编

使用重复汇编伪指令可以让汇编程序对某些语句序列进行重复汇编。重复汇编指令可以有以下三种。

1. 定重复

```
REPT    表达式
  :
  :     重复语句序列
  :
ENDM
```

表达式的值为REPT与ENDM之间的语句序列重复汇编次数

例如：

```
M=0
NUM=5
REPT 5
  M=M+1
  DB NUM*M
ENDM
```

上述语句经汇编后等效下面的语句：

DB 5，0AH，0FH，14H，19H

2. 不定重复

格式：

```
IRP   形参,<实参1,实参2,.....>
:
:      重复语句序列
ENDM
```

IRP与ENDM之间语句序列的重复汇编次数由实参的个数确定，每次重复汇编时，依次用实参表中的实参替代形参。

3. 不定重复字符

格式：

```
IRPC  形参,字符串
:     重复语句序列
:
ENDM
```

IRPC与ENDM之间的语句序列的重复次数由字符串中字符的个数来确定。每次汇编重复语句序列时，依次用字符串中一个字符替代形参。

§8.5 条件汇编

使用条件汇编，可以使宏汇编语言源程序中某些部分，在汇编期间按照给定条件产生目标代码或不产生目标代码。

基本格式：

```
IF  XX   表达式
    :
    :    条件块1
ELSE
    :
    :    条件块2
ENDIF
```

其中：XX为指定的条件，如果在汇编时，指定的条件成立，则将条件块1的语句序列汇编成相应的目标代码，否则汇编条件块2。

ELSE及相应的条件块2为可选项。

除了上述基本的条件汇编伪指令外， MASM还提供其它的条件汇编伪指令。如下表所示。

伪　指　令	汇　编　条　件
IF　1	在第一遍扫描时,扫描条件块语句序列
IF　2	在第二遍扫描时,扫描条件块语句序列
IF　表达式	表达式≠0
IFE　表达式	表达式=0
IFDEF　符号	符号已定义或被说明为 EXTRN
IFNDEF　符号	符号未定义或未被说明为 EXTRN
IFB　变量	变量是空格
IFNB　变量	变量不是空格
IFIDN　变量 1,变量 2	变量 1 与变量 2 的字符串相同
IFNIDN　变量 1,变量 2	变量 1 与变量 2 的字符串不相同