第四章 8086汇编语言程序设计

- 几个概念
- 8086汇编语言的语句
- 8086汇编中的伪指令
- 8086汇编中的运算符
- 汇编语言程序设计
- 宏定义与宏调用
- 系统调用

几个概念

- 汇编语言
- 汇编语言源程序
- 汇编
- 汇编程序

4．1　8086汇编语言的语句

汇编语言由指令性语句和指令性语句组成

- 一、指令性语句格式

  [标号：] 操作码 [操作数1，] [操作数2] ；[注释]

- 二、指示性语句格式

  [标识符（名字）] 指示符（伪指令） 表达式

- 三、有关属性

  存储器操作数的属性有三种：段值、段内偏移量和类型。

4．2　8086汇编中的伪指令

- 一、 符号定义语句
  - 1、等值语句

    格式：符号名　EQU　表达式

    例：　⑴ PORT　　EQU　1234

    ⑵ BUFF　　EQU　PORT＋58

    ⑶ MEM　　EQU　DS:[BP＋20H]

    ⑷ COUNT　EQU　CX

    ⑸ ABC　　EQU　AAA

2、等号语句

格式：　　NUM＝34

……

NUM＝34＋1

- 二、变量定义语句

格式： 符号名 DB/DW/DD 表达式

- 1、定义一组数据

  例1： BUFF DW 1234H, 0ABCDH, 8EH

  DW –79DH, 7B6AH

- 2、定义一串字符

  例2： STR DB ‘Welcome !’

- 3、定义保留存储单元

  例3： SUM DW ？，？

- 4、复制操作

  复制操作符DUP（Duplication）可预置重复的数值

  例4： ALL_ZERO DB 0，0，0，0，0

  用复制操作可改为：

  ALL_ZERO DB 5 DUP（0）

- 5、将已定义的地址存入内存单元

  例5： LIT DD CYC

  …

  CYC: MOV AX , BX

- 三、段定义语句
  - 1、 段定义语句格式：

段名 SEGMENT [定位类型] [组合类型] [‘类别’]

……

段名 ENDS

只需要掌握

段名 SEGMENT

……

段名 ENDS

- 2、段假设语句

  ASSUME 段寄存器名：段名[,…]

- 3、 ORG伪指令、地址计数器 $

  ORG伪指令格式： ORG <表达式>

- 4、END 表示源代码结束

  格式为： END 常数或表达式.

- 四、过程定义语句

过程名 PROC NEAR/FAR

……

RET

过程名 ENDP

4．3 8086汇编中的运算符

一、常用运算符

1、算术运算符

＋（加）、－（减）、*（乘）、/（除）、MOD（求余）。

2、逻辑运算符

AND、OR、XOR、NOT

3、关系运算符

EQ（相等）、NE（不等、）
LT（小于）、GT（大于）、
LE（小于等于）、GE（大于等于）。
关系成立，全1
关系不成立，全0

例： MOV DL , 10H LT 16

→ MOV DL,0

例： AND AX , 555 GT 222

→ AND AX,0FFFFH

二、 常用的操作符

4、分析操作符

1）SEG操作符

例： MOV AX , SEG BUFF

2）OFFSET操作符

例：MOV BX , OFFSET BUFF

3）TYPE操作符

对于变量有3种： 1____字节型；

2____字型；

4 ____双字型；

对于标号有2种：－1____NEAR（段内），

－2 ____FAR（段间）。

例： BUFF DB 20H

MOV BX ,TYPE BUFF ⇒

MOV BX,01

4）LENGTH操作符

5）SIZE操作符

SIZE＝TYPE×LENGTH

例：**BUFF DB 10DIP(?)**

MOV CX,LENGTH BUF

⇒ **MOV CX,10**

MOV CX,SIZE BUF

⇒ **MOV CX,20**

5、综合运算符（合成操作符）

功能：

1、由已有的操作数建立新的操作数；

2、新操作数和已有的操作数类型不一样

1）PTR运算符

格式：　类型　PTR　表达式

例：**INC WORD PTR [BX]**

```
wx      dw 1234h
mov al, byte ptr wx
mov ah, byte ptr wx+1
```

2）THIS操作符

格式：　THIS　类型(或属性)

例：**FIRST　EQU THIS BYTE**

SECOND DW　100　DUP （?）

4．4汇编语言程序设计

常用的汇编语言程序框架：

```
DATA    SEGMENT                 ；定义数据段
VAL1    DB  12H , 8EH           ；定义变量
        ……
DATA    ENDS                    ；数据段结束

CODE    SEGMENT                 ；定义代码段
        ASSUME  DS:DATA , CS: CODE          ；段属性说明
START: MOV  AX , DATA           ；初始化DS
        MOV  DS , AX
        ……
        MOV  AX , 4C00H         ；返回DOS
        INT  21H
CODE    ENDS                    ；代码段结束
        END  START              ；源程序结束
```

- 特点
  - 程序分段
  - 语句由指令性和指示性语句组成
  - 两种程序框架

框架一：

```
DATA    SEGMENT                  ；定义数据段
VAL1    DB  12H , 8EH            ；定义变量
        ……
DATA    ENDS                     ；数据段结束

CODE    SEGMENT                  ；定义代码段
        ASSUME  DS:DATA , CS: CODE          ；段属性说明
START: MOV  AX , DATA            ；初始化DS
        MOV  DS , AX
        ……                       ；填写代码
        MOV  AX , 4C00H          ；返回DOS
        INT  21H
CODE    ENDS                     ；代码段结束
        END  START               ；源程序结束
```

框架二：

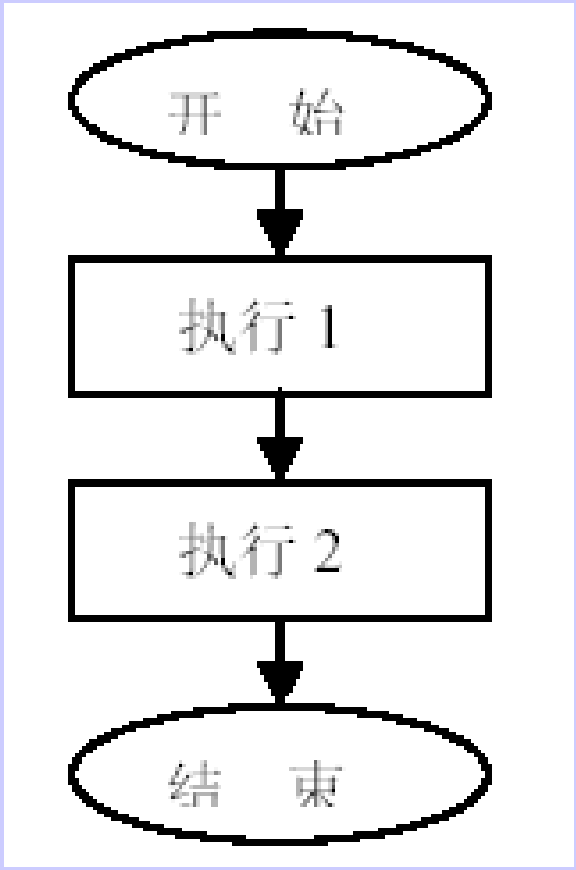
```
DATA    SEGMENT                  ；定义数据段
VAL1    DB  12H , 8EH            ；定义变量
        ……
DATA    ENDS                     ；数据段结束

CODE    SEGMENT                  ；定义代码段
MAIN    PROC FAR
        ASSUME  DS:DATA , CS: CODE          ；段属性说明
START:  PUSH DS
        MOV  AX , 0
        PUSH AX
        MOV AX,DATA
        MOV DS,AX
        ……                       ；填写代码
        RET
MAIN    ENDP
CODE    ENDS                     ；代码段结束
        END  START               ；源程序结束
```

一、汇编语言程序设计基本步骤

1. 分析问题，确定模型→
2. 确定算法→
3. 绘制流程图→
4. 编写程序→
5. 检查和调试

二、汇编语言程序的基本结构

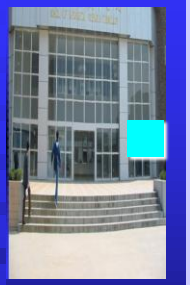
1、顺序结构



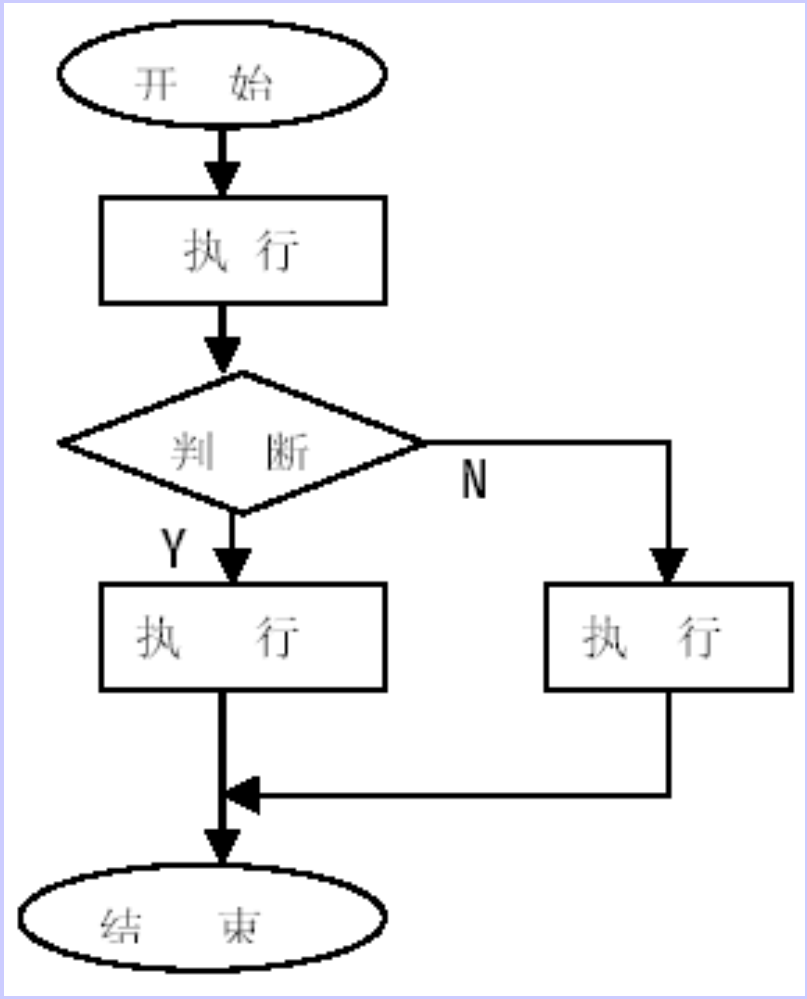
举例

- 把al内容除2，并四舍五入
- 完成

  x1*x2→x3　x1:byte　x2:word　x3 34bit



2、分支结构



举例

- 实现

$$y=\begin{cases}-1 & x<0\\ 0 & x=0\\ 1 & x>0\end{cases}$$

- 用查表法确定分支

  a、对8种产品进行编号0，1，……7

  b、每一个编号对应一个入口地址

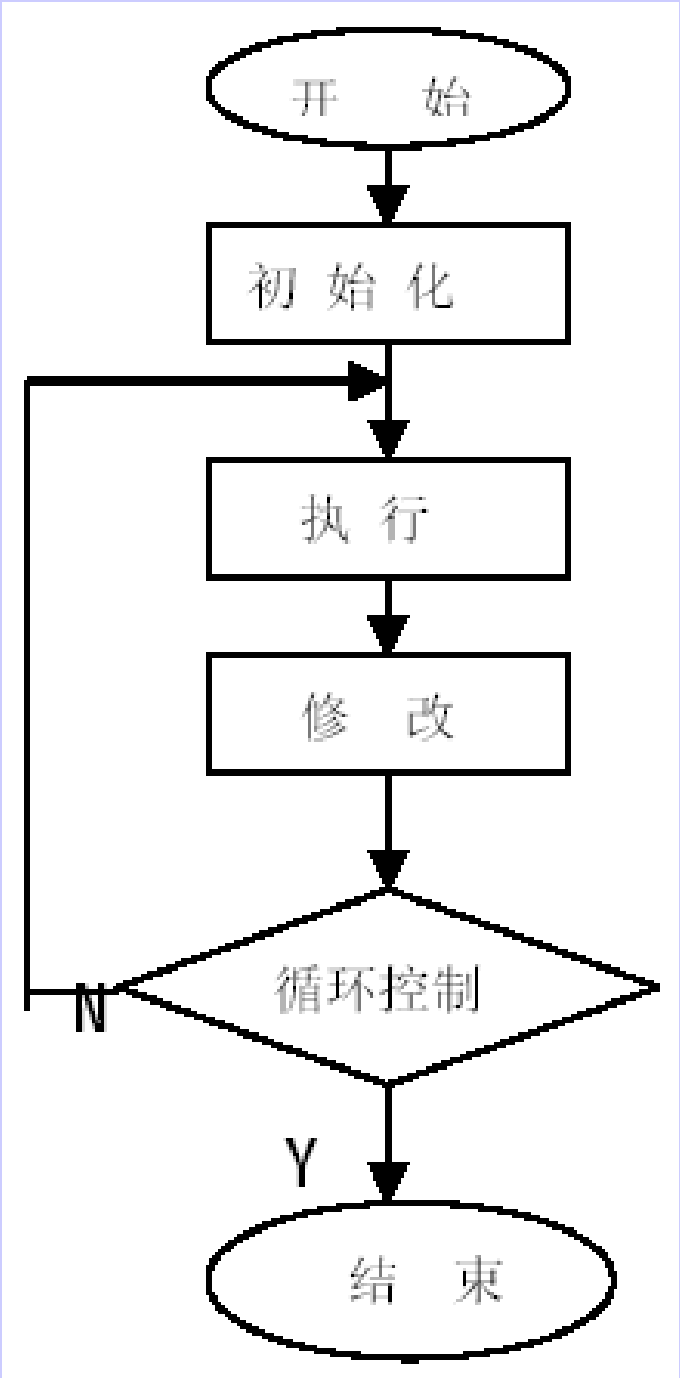
  proc0，proc1，……proc7

  步骤：

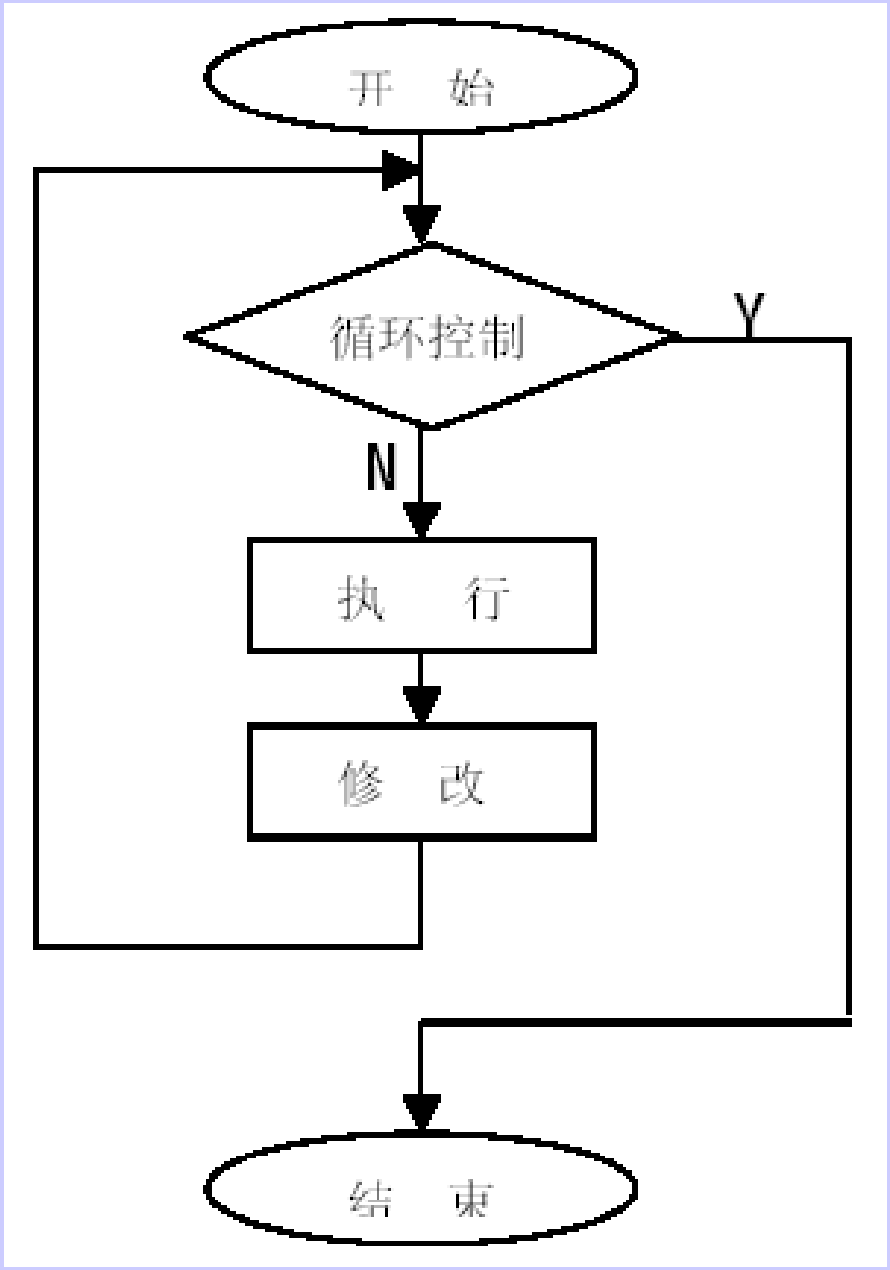
  a、把入口地址放在Table开始的内存单元；

  b、根据key内容进行查表实现分支转移

3、循环结构



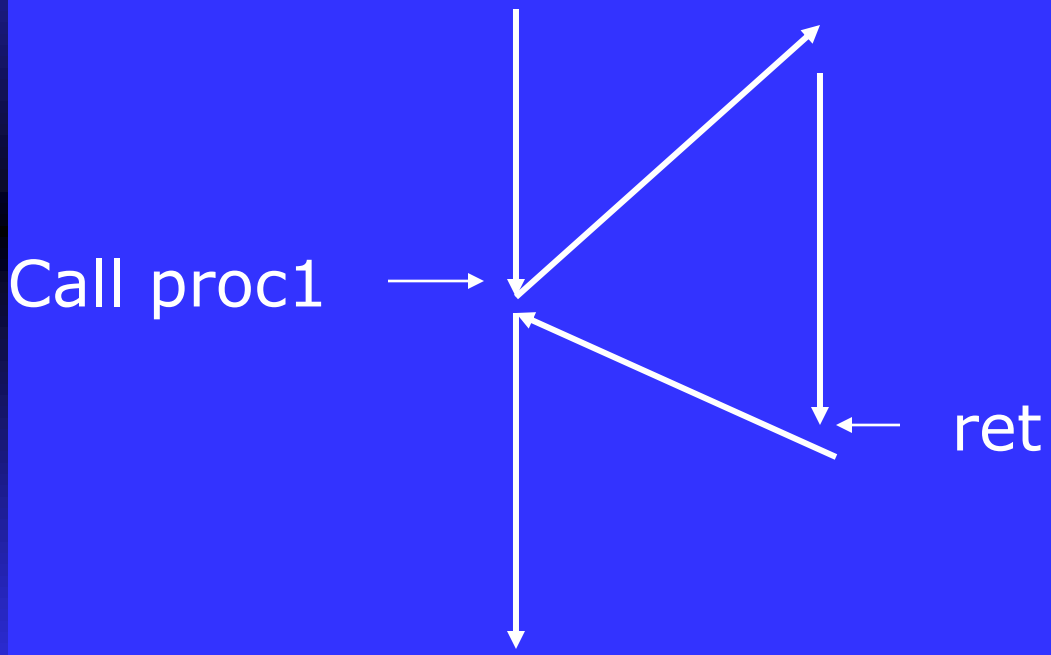

USTC


开　始
循环控制
Y
N
执　行
修　改
结　束

举例

- 例1、计算y＝0＋2＋4＋……＋1998
  y→dxax
- 查找数组中是否由关键字，
  有：变量dd1←1；dd2←关键字偏移量
  无：变量dd1←0
  - 方法一、用loop指令
  - 方法二、用loopnz（loopne）指令


USTC


Call proc1
ret

- 实现方法
  - 主程序有调用指令call
  - 子程序有返回指令ret
- 特点
  - Call指令与ret缺一不可
  - 现场保护
  - 入口出口参数设置

- 举例

  编写AX*10→AX子程序

- 子程序编写
  - 1、子程序组成
    - 使用说明
      - a、子程序功能
      - b、使用寄存器情况
      - c、入口出口参数
    - 程序体
      - 入口出口参数传递
      - 现场保护
      - 功能程序段


```
                ；这是一个乘10子程序
                ；使用寄存器AX
                ；入口：AX，出口：AX
mul10           proc far
                push bx             ;保护现场
                pushf
                add ax,ax           ;2ax
                mov bx,ax           ;2ax→bx
                add ax,ax           ;4ax
                add ax,ax           ;8ax
                add ax,bx           ;10ax
                popf                ；恢复现场
                pop bx
                ret
mul10           endp
```

功能程序段

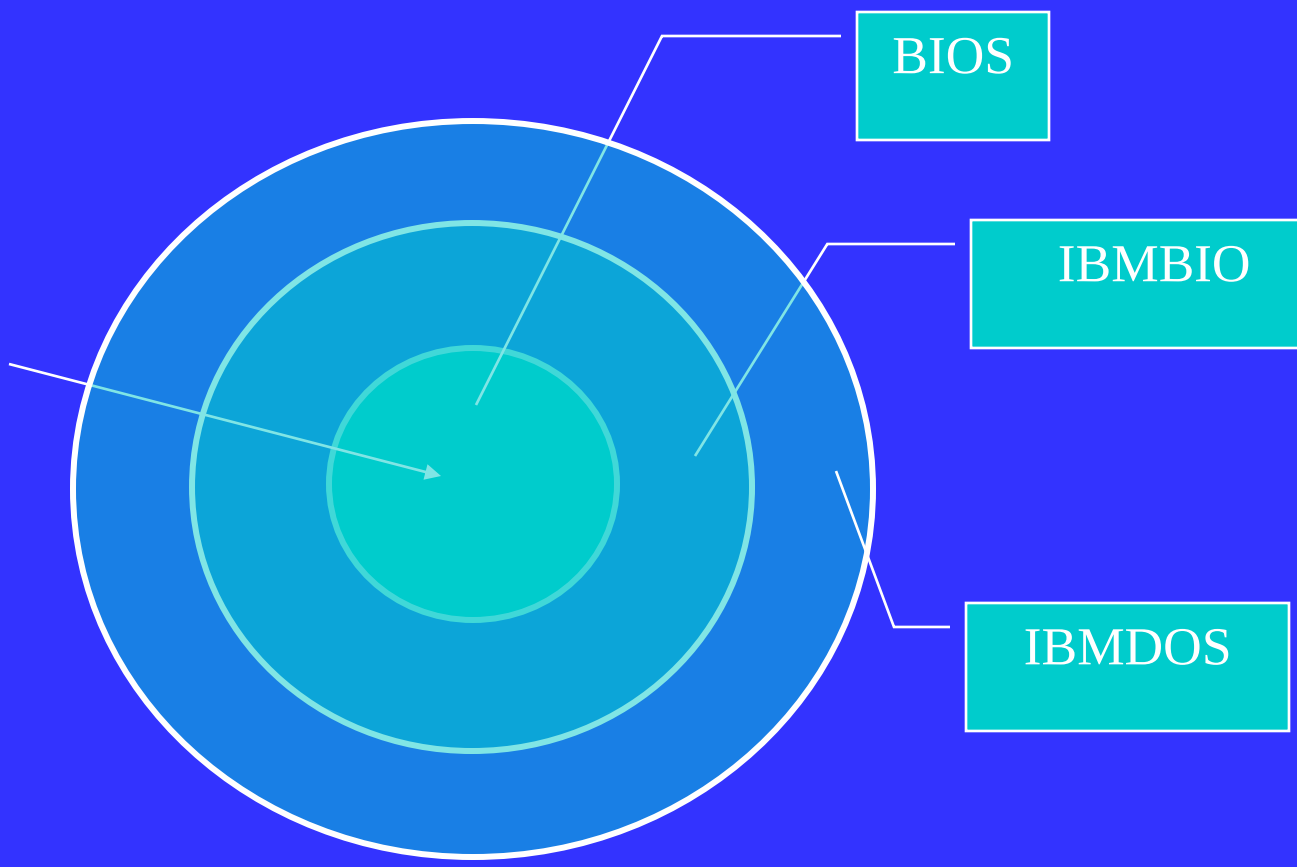
- 2、现场保护和入口出口参数传递
  - （1）、现场保护
    - 方案一：在子程序中保护
      - 灵活，不方便
    - 方案二：在主程序中保护
      - 方便，不灵活
  - （2）、参数传递
    - 寄存器传递
    - 变量传递
    - 堆栈传递

- 举例
  - 1、数组元素求和

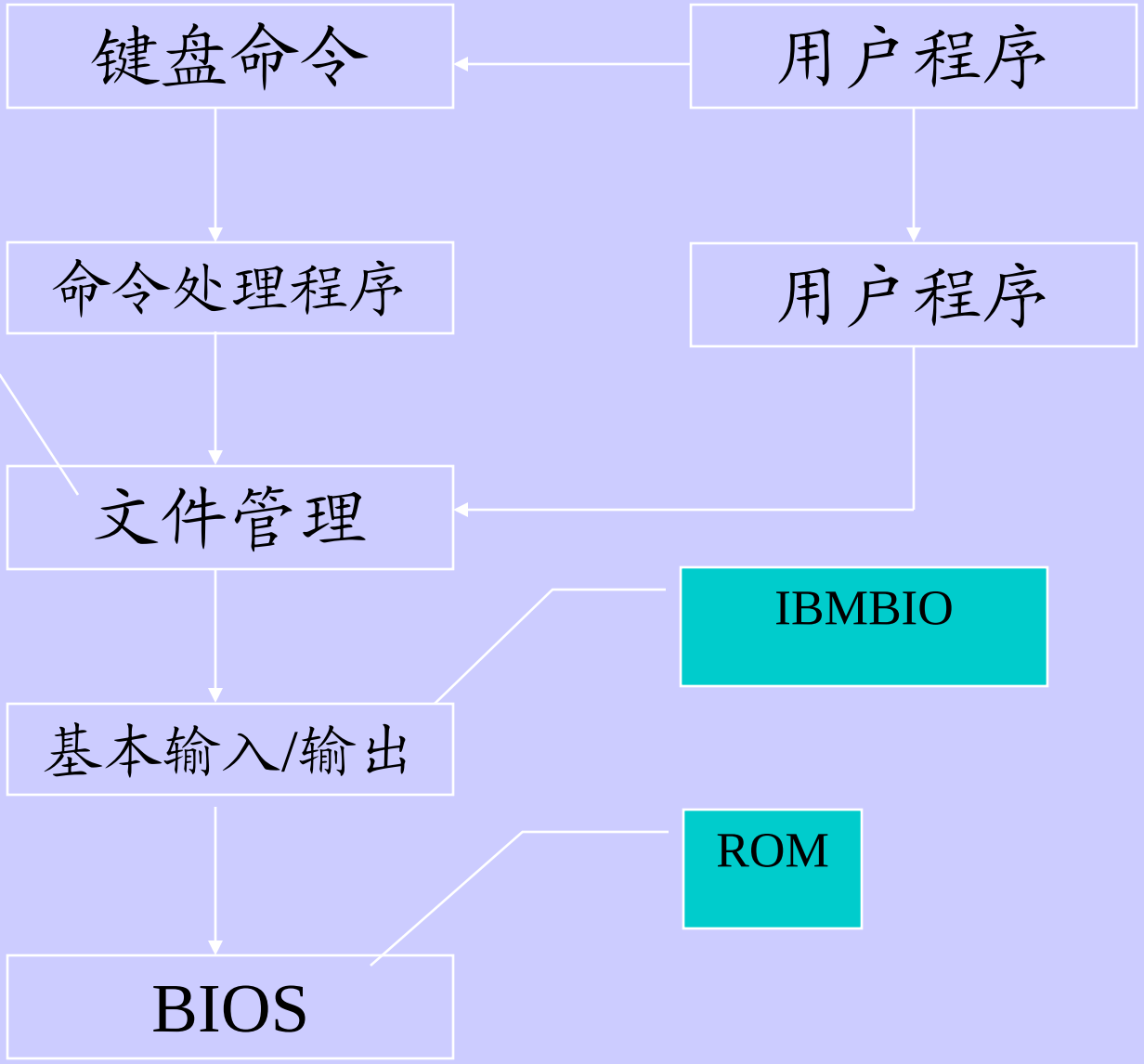
- 2、计算 N！
  - （1）如果AL＝0，则结果为1；
  - （2）如果AL！＝0，在堆栈中形成n，n－1，n－2，……1；
  - （3）从堆栈中推出1，2，…n－1，n－2，计算n！

4.4 系统调用

- 一、PCDOS执行流程


BIOS
IBMBIO
IBMDOS


USTC


键盘命令
用户程序
命令处理程序
用户程序
IBMDOS
文件管理
IBMBIO
基本输入/输出
ROM
BIOS

- 二、DOS系统子程序功能
  - 1、磁盘管理
  - 2、内存管理
  - 3、系统输入/输出

  共87个子程序

- 三、系统功能调用

  对87个子程序进行编号，根据编号进行相关处理

  格式

  1、入口参数

  2、编号→ah

  3、INT 21H

编号→ah

INT 21

ah=1　　ah=2

1号调用　2号调用

- 常用的系统调用
  - 1、1号调用－键盘输入
  - 2、8号调用－键盘输入
  - 3、6号调用－显示及输入
  - 4、9号调用－输出字符
  - 5、10号调用－输入字符串

4.5 宏定义与宏调用

- 一、宏定义和宏调用
  - 1、格式

    宏指令名 macro [参数表]

    …　　;宏定义体

    endm
  - 2、优点
    - 缩短源程序长度，但不减少代码长度；
    - 程序易读

- 3、参数可以是多种形式

- 二、宏定义取消伪操作

  格式

  purge 宏命令名1，宏命令名2，…

- 三、重复伪操作

  格式

  rept <表达式>

  ；重复块

  endm

- 四、条件汇编

  格式

  if　表达式

  …

  else

  …

  endif