

汇编语言程序设计

源文件包含和模块连接



多模块程序结构

- 程序分段、子程序等实现了程序模块化
- 开发大型应用程序时常使用
 - ▶ 多个源程序文件
 - ▶ 目标代码模块等
- 组成完整的程序
形成多模块程序结构

MASM支持的多模块方法

- ✓ 源文件包含
- ✓ 模块连接
- ✓ 子程序库
- ✓ 库文件包含

源文件包含

- 大型源程序可以合理地分放在若干个文本文件中
 - ▶ 各种常量定义、声明语句等组织在包含文件 (*.INC)
 - ▶ 常用的或有价值的宏定义存放在宏定义文件 (*.MAC)
 - ▶ 常用的子程序形成汇编语言源文件 (*.ASM)
 - ▶ 任何文本文件

➤ 使用源文件包含伪指令 INCLUDE

include 文件名

; 将指定文件内容插入主体源程序文件

存储器数据显示程序—1

;文件名: **eg0508.inc**, 程序的数据段内容

.data

...

;文件名: **eg0508s.asm**, 子程序

rdhd proc ;十六进制输入子程序

...

dphd proc ;十六进制输出子程序

...

write proc ;显示有符号十进制数子程序

...

存储器数据显示程序—2

;文件名: eg0508.asm, 主程序

include io32.inc ;包含软件I/O库声明

include eg0508.inc ;包含数据段

.code ;代码段, 主程序

...

exit 0 ;主程序结束

include eg0508s.asm ;包含子程序代码

end start



源文件包含的使用

➤ 被包含文件

- ▶ 文件名要符合操作系统规范
 - ▶ 只能是文本文件
 - ▶ 内容被插入源文件包含include语句所在的位置
- ## ➤ 实质仍然是一个源程序
- ▶ 只是分开在若干个文件中
 - ▶ 只需针对主体源程序文件进行汇编、连接

操作过程

```
make32 eg0508
```



多模块程序结构

- 程序分段、子程序等实现了程序模块化
- 开发大型应用程序时常使用
 - ▶ 多个源程序文件
 - ▶ 目标代码模块等
- 组成完整的程序
形成多模块程序结构

MASM支持的多模块方法

- ✓ 源文件包含
- ✓ 模块连接
- ✓ 子程序库
- ✓ 库文件包含

模块连接

- 子程序单独编写一个源程序文件
 - ▶ 使用共用伪指令PUBLIC和外部伪指令EXTERN声明
 - ▶ 子程序在代码段，与主程序文件采用相同的存储模型
 - ▶ 没有开始执行和结束执行点，但有汇编结束语句
 - ▶ 处理好子程序与主程序之间的参数传递问题
- 子程序源文件
- 连接时输入子

;定义标识符的模块使用

public 标识符 [,标识符 ...]

;调用标识符的模块使用

extern 标识符:类型 [,标识符:类型 ...]

存储器数据显示程序—3

;文件名: eg0508es.asm, 子程序

public rdhd,dphd,write ;子程序共用

extern temp:dword ;外部变量

rdhd proc ;十六进制输入子程序

...

end ;汇编结束



存储器数据显示程序—4

;文件名: eg0508e.asm, 主程序

...

temp dword ? ;共享变量定义

extern rdhd:near,dphd:near,write:near ;外部子程序

public temp ;变量共用

.code ;代码段, 主程序

...

end start



模块连接的操作过程

- 子程序单独编写一个源程序文件
- 子程序源文件汇编形成目标模块OBJ文件
- 连接时输入子程序目标模块文件名

操作过程

```
ML /c /coff eg0508e.asm
```

```
ML /c /coff eg0508es.asm
```

```
LINK32 /subsystem:console eg0508e.obj eg0508es.obj
```



汇编语言程序设计

子程序库和库文件包含



多模块程序结构

- 程序分段、子程序等实现了程序模块化
- 开发大型应用程序时常使用
 - ▶ 多个源程序文件
 - ▶ 目标代码模块等
- 组成完整的程序
形成多模块程序结构

MASM支持的多模块方法

- ✓ 源文件包含
- ✓ 模块连接
- ✓ 子程序库
- ✓ 库文件包含

子程序库

- 子程序库是子程序模块的集合，便于统一管理子程序
- 编写存入库文件的子程序
 - ▶ 遵循更加严格的子程序模块要求
 - ▶ 应该遵循一致的规则（以免在使用时造成混乱）
- 子程序文件编写完成、汇编形成目标模块
- 利用库管理工具程序把子程序模块加入到子程序库



子程序库的使用

- 子程序单独编写一个源程序文件
- 子程序源文件汇编形成目标模块OBJ文件
- 利用库管理工具把子程序模块加入到子程序库
- 在连接主程序时提供子程序库文件名

操作过程

```
ML /c /coff eg0508e.asm
```

```
ML /c /coff eg0508es.asm
```

```
LIB32 /OUT:eg0508.lib eg0508es.obj
```

```
LINK32 /subsystem:console eg0508e.obj eg0508.lib
```

多模块程序结构

- 程序分段、子程序等实现了程序模块化
- 开发大型应用程序时常使用
 - ▶ 多个源程序文件
 - ▶ 目标代码模块等
- 组成完整的程序
形成多模块程序结构

MASM支持的多模块方法

- ✓ 源文件包含
- ✓ 模块连接
- ✓ 子程序库
- ✓ 库文件包含



库文件包含

- 要使用已存入库文件中的子程序
- 在主程序源文件中用库文件包含伪指令 INCLUDELIB 声明

➤ 使用库文件包含伪指令 INCLUDELIB

includelib 文件名

;使用库文件中的子程序



库文件包含的使用

- 将子程序源文件汇编、模块文件加入子程序库

```
ML /c /coff eg0508es.asm
```

```
LIB32 /OUT:eg0508.lib eg0508es.obj
```

- 源文件中用库文件包含伪指令INCLUDELIB声明
- 正常对主程序汇编、连接，无需在连接时输入库文件名

- ▶ 编写主程序、子程序更加独立
- ▶ 子程序使用更方便

操作过程

```
make32 eg0508e
```

IO32.INC

- 组合源文件包含和库文件包含等方法
 - ▶ 可以精简程序框架，简化程序设计

```
;declare procedures
extern readc:near, readmsg:near
extern dispc:near, dispmsg:near, dispctrl:near
    ...
;declare I/O libraries
includelib io32.lib
```

```
;源程序框架
include io32.inc
    ...
```

汇编语言程序设计

宏汇编



宏汇编

- 宏（Macro）是具有宏名的一段汇编语句序列
- 宏需要先使用MACRO/ENDM伪指令进行定义
- 然后在程序中使用宏名（带参数）进行宏调用
- 源程序进行汇编时

宏名被汇编程序用宏定义的
代码序列替代，实现宏展开

- 这个过程就是“宏汇编”

➤宏定义

宏名 **macro** [形参表]
 ... ;宏定义体
 endm



宏定义、宏调用和宏展开

;宏定义

```
WriteString macro msg  
    push eax  
    lea eax,msg  
    call dispmsg  
    pop eax  
endm
```

;宏调用（宏指令）

```
WriteString msg
```

;宏展开

```
    push eax  
    lea eax,msg  
    call dispmsg  
    pop eax
```

```
msg byte 'Hello, Assembly !',13,10,0
```



宏汇编的特点

- 宏需要先定义后使用，且不必在任何段中
 - ▶ 常书写于源程序开始位置
 - ▶ 常用的宏定义可以单独写成一个宏定义文件
- 宏定义中更改了寄存器内容，最好进行保护和恢复
- 宏定义的参数灵活，宏定义允许嵌套和递归调用
- 宏调用不需要控制的转移与返回
 - ▶ 宏调用将相应的语句序列复制到宏指令的位置
 - ▶ 宏展开被嵌入源程序，成为一体



宏与子程序：简化程序

➤ 宏仅是源程序级的简化

- ▶ 宏调用在汇编时进行程序语句的展开，不需要返回
- ▶ 不减小目标程序，执行速度没有改变

➤ 子程序不仅简化源程序，还是目标程序级的简化

- ▶ 子程序调用在执行时由CALL指令转向、RET指令返回
- ▶ 形成的目标代码较短，执行速度减慢



宏与子程序：传递参数

- 宏通过形参、实参结合实现参数传递
 - ▶ 使用软件方法，简洁直观、灵活多变
 - ▶ 传递出错多体现为语法错误，易于发现
- 子程序利用寄存器、存储单元或堆栈等实现
 - ▶ 运用硬件本身，规则严格、方法固定
 - ▶ 传递出错常反映为逻辑或运行错误，较难排除



宏与子程序：选用原则

- ▶ 当程序段较短或要求较快执行时

➤ 选用宏

- ▶ 宏常依附于源程序，适合进行全局性预处理

- ▶ 当程序段较长或为减小目标代码时

➤ 选用子程序

- ▶ 子程序更具有独立性，可以分别编写

