

# C3 汇编语言程序实例及上机操作

- ❖ 汇编语言的工作环境 >>
- ❖ 汇编语言程序实例 >>
- ❖ 程序实例的上机步骤 >>
- ❖ 在**Win7**系统中执行汇编 >> 部分内容提前介绍
- ❖ 几个常用的**DOS**系统功能调用 >>

# ← 汇编语言的工作环境

---

- ❖ 汇编语言的系统工作文件
- ❖ 进入**DOS**命令行的方式
- ❖ 常用的**DOS**命令

# 汇编语言的系统工作文件

- ❖ 使用编辑程序**编辑**源程序文件（.asm）
- ❖ 使用汇编程序（MASM）将源程序文件（.asm）**汇编**成目标文件（.obj）
- ❖ 使用连接程序（LINK），将目标文件（.obj）**连接**成可执行文件（.EXE）
- ❖ 使用调试程序（DEBUG），调试可执行文件

编辑  
→

EDIT

MY.ASM  
文件

汇编  
→

MASM

MY.OBJ  
文件

连接  
→

LINK

MY.EXE  
文件

# 运行汇编语言程序需要以下文件

---

- ❖ 编辑程序 **EDIT.COM**
- ❖ 汇编程序 **MASM.EXE**
- ❖ 连接程序 **LINK.EXE**
- ❖ 调试程序 **DEBUG.EXE**

# 注意几点

---

## ❖ 本课程使用

- **Microsoft Masm 6.15, Microsoft LINK 5.13**版本。

## ❖ **EDIT.COM** 和 **DEBUG.EXE**为系统自带。

## ❖ 为方便操作，系统文件和用户文件尽可能放在同一文件目录下。


## ← DosBox环境

---

- ❖ 对于初学者，使用**DosBox**是一个较好的64位环境下编译汇编程序的解决方案。
- ❖ 下载安装**DosBox**，安装目录缺省为 **C:\Program Files (x86)\DOSBox-0.74**。

- ❖ DosBox为Windows环境下的**Dos模拟器**
- ❖ 可以将Dos程序放置在该环境中运行。其过程即为挂载。挂载命令为**mount**。
- ❖ 这里需要挂载的Dos程序为汇编编译程序**MASM 6.15**，如图所示。在DosBox环境下Z:>提示符下键入命令
  - **mount C: D:\MASM6.15**
- ❖ 如果挂载成功，会在该命令的下面出现提示语句：
  - **Drive C is mounted as local directory D:\MASM6.15\**





---

❖ 在**DosBox**环境下，查看**C:**目录下的文件，可以看出和**Windows**下**D:\MASM6.15**的文件完全一样。即**mount**命令将原本存放在**Windows**环境下的文件映射到了**DosBox**中。

# 执行汇编



## ❖ 注意：

- 如果在实际存放目录（例如 **D:\MASE6.15**）中新存放一个事先写好的汇编程序，在**DosBox**中需要重新启动并进行挂载，才能对该汇编程序进行debug等相关操作。

## ← 3.2 汇编语言程序实例

---

❖ 实例1

❖ 实例2

# 例 单个字符的键盘输入与显示输出的程序

```
code segment
    assume cs:code
start: mov ah,1
       int 21h
       mov dl,al
       add dl,1
       mov ah,2
       int 21h
       mov ah,4ch
       int 21h
code ends
       end start
```

运行结果：键入A接着显示B，键入K接着显示L

## 例3.2 显示“HELLO, WORLD!”的程序

```
data segment
    string db 'HELLO, WORLD! $'
data ends
code segment
    assume cs:code,ds:data
start: mov ax,data
        mov ds,ax
        mov dx,offset string
        mov ah,9
        int 21h
        mov ah,4ch
        int 21h
code ends
    end start
```

## ← 3.3 程序实例的上机步骤

---

- ❖ 3.3.1 编辑 -- 建立ASM源程序文件
- ❖ 3.3.2 汇编 -- 产生OBJ二进制目标文件
- ❖ 3.3.3 连接 -- 产生EXE可执行文件
- ❖ 3.3.4 关于LST列表文件
- ❖ 3.3.5 程序的运行和调试

## 3.3.1 编辑建立ASM源程序文件

1. 进入DOS命令行方式。

2. 假定汇编语言的系统工作文件目录为D:\MASM6.15\，其中D:\表示D盘的根目录。可以通过以下命令指向D盘：

D: ✓

3.如果屏幕显示不在此目录，可以通过以下命令进入该目录：

D:\>CD \MASM6.15 ✓

注意，加黑字体是键入的命令。当屏幕显示进入该目录后，用如下命令编辑源程序文件：

D: \>MASM6.15\>EDIT HELLO.ASM ✓

## 3.3.2 汇编 -- 产生OBJ二进制目标文件

- ❖ 假定汇编语言源程序文件 **HELLO.ASM** 已经在当前目录 **D:\MASM6.15\** 下，用如下命令进行汇编：
  - **D:\>MASM6.15\>MASM HELLO ✓**
- ❖ 该命令执行后，将产生一个同名的二进制目标文件 **HELLO.OBJ**。下一步就是对这个 **HELLO.OBJ** 文件进行连接以产生最后的可执行文件。
- ❖ 如果源程序有语法错误，则不会产生目标文件。同时报错，提示源程序的出错位置和错误原因。



### 3.3.3 连接产生EXE可执行文件

- ❖ 使用连接程序LINK把目标文件(OBJ)转换为可执行的EXE文件。键入以下命令：
  - D: \>MASM6.15\>LINK HELLO ✓
- ❖ 因为源程序中没有定义堆栈段，所以连接程序给出无堆栈段的警告，其实并不是错误，并不影响程序的运行。到此为止，连接过程已经结束。

## 3.3.4 关于LST列表文件

❖ 同时得到obj和lst文件的命令

- D: \>MASM6.15\>MASM HELLO HELLO HELLO ✓

❖ 列表文件报告了汇编过程中产生的很多有价值的参考信息。主要包括

- 源程序和机器语言清单
- 指令和变量的偏移地址等等。

## 3.3.5 程序的运行

❖ 建立了EXE文件后，就可以直接在DOS的提示符下，输入EXE文件的文件名，如：

D>Hello↓

直接运行程序。

❖ 对EXE文件无需扩展名就可执行。

### ← 3.1.3 常用的DOS命令

1. 盘： ； 选择盘符

如果屏幕显示为C:\>， 表示你当前在C盘， 你希望到E盘， 则可键入：

C:\>E: ↵ (↵ 表示Enter键)

## 3.1.3 常用的DOS命令

2. **CD** ; 选择目录

例如:

E:\>**CD** ; 显示当前目录, 当前目录是根目录

E:\>**CD MASM** ; 进到MASM子目录,

E:\>MASM>**CD MY** ; 从当前目录MASM进到下一级MY子目录

E:\>MASM\MY>**CD..** ; 从当前目录MY退到上一级目录MASM

E:\>MASM>**CD\** ; 从当前目录MASM退到根目录E:\>

## 3.1.3 常用的DOS命令

3. DIR ; 显示目录和文件

例如:

**E:\>MASM>DIR** ; 列出当前目录下的子目录和文件

**E:\>MASM\>DIR \*.ASM** ; 列出所有扩展名为ASM的文件, \*为通配符

**E:\>MASM>DIR HELLO.\*** ; 列出所有名为HELLO而扩展名不限的文件

**E:\>MASM>DIR HE\*.???** ; 列出所有文件名前2个字符为'HE'而扩展名有3个字符的文件

### 3.1.3 常用的DOS命令

---

4. **REN** ; 改变文件名

例如:

**E:\>REN H1.TXT H2.ASM** ; 把文件**H1.TXT**改  
名为**H1.ASM**



## 3.1.3 常用的DOS命令

### ❖ 5. CLS

； 清除屏幕

### ❖ 6. DEL

； 删除文件

- 例如：

- E:\>DEL C.TXT

； 删除文件C.TXT

### ❖ 7. MD

； 建立目录

- 例如：

- E:\>MD MASM

； 建立MASM目录



### 3.1.3 常用的DOS命令

❖ 8. **RD** ; 删除目录

- 例如:

- `E:\>MASM\>RD ASM` ; 删除下级子目录ASM

## 3.1.3 常用的DOS命令

### ❖ 9. COPY ; 复制文件

- 例如:
- E:\>COPY H1.TXT H2.TXT ; 复制文件H1.TXT到文件H2.TXT
- E:\>COPY A+B C.TXT ; 把文件A和B连接后得到文件C.TXT

### 3.1.3 常用的DOS命令

#### ❖ 10. **TYPE** 显示文本文件的内容

- 例如:

- **E:\>TYPE C.TXT** ; 显示文件C.TXT的内容

#### ❖ 11. **>** ; 输出的重定向操作符

- 例如:

- **E:\>DIR >THIS.TXT** ; 把DIR显示结果输出到文件  
**THIS.TXT**

## 3.1.3 常用的DOS命令

### ❖ 12. SET PATH 设置或显示可执行文件的搜索路径

#### ■ 例如:

- E:\>PATH ; 显示可执行文件的搜索路径
- E:\>SET PATH ; 显示可执行文件的搜索路径
- E:\>SET PATH=E:\MASM; D:\MASM6 ; 设置可执行文件的搜索路径
- EXE, COM, BAT都是可执行文件, 设置搜索路径后, 文件按路径搜索并执行。

## 3.1.3 常用的DOS命令

### ❖ 13. **HELP** 显示命令格式和用法

- **E:\>HELP** ; 显示所有命令的格式
- **E:\>HELP DIR** ; 显示DIR命令的用法

## 3.3.6 程序的调试

---

- ❖ 调试程序 **DEBUG.EXE** 是 **WINDOWS** 系统自带的。
- ❖ **DEBUG** Hello.EXE
- ❖ “-” **DEBUG** 命令提示符
- ❖ **READU GPT**



---

❖ **-r**

■ **-r ax**

❖ **-d 073F:0100**

❖ **-e 2000:0000 12 34 56 AB 3F F3**

# 1. 反汇编命令U

---

## ❖ 格式1: U 地址

- 地址用偏移地址或者段地址：偏移地址表示。
- 该命令从指定的地址开始，把机器语言反汇编为汇编语言。
- 若省去指定地址，则以上一个U命令反汇编的最后一条指令地址的下一个单元作为起始地址。



## 2. 运行程序命令G

❖ 格式: **G**[=起始地址][中止地址(断点)]

- 起始地址规定了执行的起始地址。
- 中止地址是断点地址, 让程序暂停在某个位置
- =不能省掉

❖ 若省略起始地址, 则以当前**CS:IP**作为起始地址。

❖ 默认段地址在段寄存器**CS**中。

### 3. 跟踪程序命令T

❖ 格式1: **T**[=起始地址]

单步执行程序，在指令执行中逐条进行跟踪，若省去地址，则从**CS: IP**现行值执行。

❖ 格式2: **T**[=起始地址][指令条数]

可对多条指令进行跟踪。

❖ 注意：对于**INT**指令不能使用**T**命令跟踪。

## 4. 单步执行程序指令P

---

- ❖ P命令，用以执行循环、重复的字符串指令、软件中断或子例程。
  - T命令无法一次执行的INT指令，P命令就可以一次执行完这个系统例行程序，回到用户程序中。

## 5. 退出命令Q

---

❖ 用Q命令退出DEBUG。

## ← 3.5 几个常用的DOS系统功能调用

### ❖ 一般程序中，涉及文本的基本IO功能有哪些？

- 键盘输入一个字符并回显
- 显示一个字符
- 显示字符串
- 键盘输入到缓冲区

### ❖ 你写的程序怎样才能具体上面这些功能？

- 汇编语言程序设计需要采用系统的各种功能程序
- **21H号中断**是DOS提供给用户的用于调用系统功能的中断，它有近百个功能供用户选择使用，主要包括设备管理、目录管理和文件管理三个方面的功能。
- 如何调用这些功能？

# 功能调用的格式

---

❖ 通常按照如下4个步骤进行：

- ① 在AH寄存器中设置系统功能调用号
- ② 在指定寄存器中设置入口参数
- ③ 执行指令INT 21H，实现中断服务程序的功能调用
- ④ 根据出口参数分析功能调用执行情况

# 1. 键盘输入一个字符并回显

---

## ❖ DOS功能调用INT 21H

- 功能号：AH=01H
- 出口参数：AL=输入字符的ASCII码
- 功能：等待从键盘输入一个字符，该字符的ASCII码送AL，并送屏幕显示。

❖ 调用此功能时，若无输入，则会一直等待，直到输入后才继续。



**mov ah,01h**; 功能号: ah←01h

**int 21h** ; 功能调用

**cmp al,'Y'** ; 处理出口参数al

**je yeskey** ; 是“Y”

**cmp al,'N'**

**je nokey** ; 是“N”

**yeskey:** ...

**nokey:**...



## 2. 显示一个字符

---

### ❖ DOS功能调用 **INT 21H**

- 功能号：AH=**02H**
- 入口参数：DL=输出字符
- 功能：在显示器当前光标位置显示给定的字符，光标右移一个字符位置。

❖ 在当前显示器光标位置显示一个问号

**mov ah,02h** ; 设置功能号: **ah**←**02h**

**mov dl,'?'** ; 提供入口参数: **dl**←**'?'**

**int 21h** ; **DOS**功能调用: 显示问号

# 3. 显示字符串

---

## ❖ DOS功能调用INT 21H

- 功能号：AH = 09H
- 入口参数：
  - DS:DX=欲显示字符串在主存中的首地址；
- 字符串应以\$（24H）结束
- 功能：显示由DS:DX指向的字符串

**string** db 'Hello,Everybody! \$' ; 在数据段定义要显示的字符串

...

mov ah,09h ; 设置功能号 ah←09h

mov dx, **offset string**; 提供入口参数

dx←字符串的偏移地址

int 21h ; DOS功能调用字符串显示

## 4. 键盘输入到缓冲区

---

### ❖ DOS功能调用INT 21H

- 功能号:  $AH=0AH$
- 入口参数:  $DS:DX$ =缓冲区首地址  
 $(DS:DX)$ =缓冲区字节数
- 功能: 输入到缓冲区

# 缓冲区的定义

Byte1	Byte2	Byte3 ...(N_max-1)
N_max	N_act	abc...

- ❖ 第1字节事先填入最多欲接收的字符个数
  - 包括回车字符，可以是1~255
- ❖ 第2字节将存放实际输入的字符个数
  - 不包括回车符
- ❖ 第3字节开始将存放输入的字符串
- ❖ 实际输入的字符数多于定义数时，多出的字符丢掉，且响铃

**buffer db 81 ; 定义缓冲区**

**; 第1个字节填入可能输入的最大字符数**

**db ? ; 存放实际输入的字符数**

**db 81 dup( ? ) ; 存放输入的字符串**

**...**

**mov dx, seg buffer ; 伪指令seg取得buffer的段地址**

**mov ds, dx ; 设置数据段DS**

**mov dx, offset buffer**

**mov ah, 0ah**

**int 21h**

## 5. 结束程序返回DOS

---

### ❖ DOS功能调用INT 21H

- 功能号：AH=4CH
- 入口参数：AL=返回码
- 功能：结束程序返回DOS

**MOV AH, 4CH**

**INT 21H**