

高等学校规划教材

微型计算机原理 及应用实验

殷代红 侯晓霞 编



化学工业出版社
教材出版中心

高等学校规划教材

微型计算机原理及应用实验

殷代红 侯晓霞 编

化学工业出版社

教材出版中心

• 北 京 •

(京)新登字 039 号

图书在版编目(CIP)数据

微型计算机原理及应用实验 / 殷代红, 侯晓霞编.
北京: 化学工业出版社, 2003.5
高等学校规划教材
ISBN 7-5025-4242-6

I. 微… II. ①殷… ②侯… III. 微型计算机-
高等学校-教材 IV. TP36

中国版本图书馆 CIP 数据核字(2003)第 030871 号

高等学校规划教材
微型计算机原理及应用实验

殷代红 侯晓霞 编

责任编辑: 唐旭华

文字编辑: 张燕文

责任校对: 洪雅妹 王素芹

封面设计: 潘 峰

*

化学工业出版社
教材出版中心 出版发行

(北京市朝阳区惠新里 3 号 邮政编码 100029)

发行电话: (010) 64982530

<http://www.cip.com.cn>

*

新华书店北京发行所经销
北京市彩桥印刷厂印刷
三河市宇新装订厂装订

开本 787 毫米×1092 毫米 1/16 印张 8 $\frac{3}{4}$ 字数 210 千字

2003 年 6 月第 1 版 2003 年 6 月北京第 1 次印刷

ISBN 7-5025-4242-6/G · 1107

定 价: 15.00 元

版权所有 违者必究

该书如有缺页、倒页、脱页者, 本社发行部负责退换

前 言

本书主要面向汇编语言程序设计、微型计算机原理及应用、微型计算机接口技术等课程，以充实、巩固课程学习，培养科学实验技能和严谨的工作作风为目的而编写的。

全书共分三个部分。第一部分为准备阶段，引导读者学习掌握上机操作过程及学会调试工具的使用。第二部分为软件实验，由指令系统与汇编语言程序设计两部分组成。各部分实验的编写出发点和指导思想有所不同，有验证性为主的实验和设计性为主的实验。从简单指令入手，熟悉和掌握 80X86 的指令系统和 IBM-PC 机的系统功能调用；从程序基本结构入手，由浅入深；从给出程序和补充完整程序进行调试到自编程序上机调试通过，逐步掌握编程方法和调试方法。通过这些实验有助于学习和掌握汇编语言程序设计的基本方法和技巧，训练学生应用 80X86 指令系统设计和编写汇编语言程序的能力。第三部分为硬件实验，由常用可编程接口芯片应用（基础性实验）和多片可编程接口芯片综合应用（提高性实验）两部分组成。在这里，有设计性为主的实验和研究探索性为主的实验。通过完成可编程芯片硬件连线与芯片的初始化编程和应用程序的设计，主要使学生掌握使用微机接口常用外围芯片的方法，培养学生使用常用可编程接口芯片并结合数字电子技术进行 I/O 接口设计的能力。附录部分包括指令表及伪指令助记符分类表，宏汇编错误信息表，DEBUG 调试程序命令表，常用芯片端子图等内容。

本书的内容具有基础性、实用性和提高性，为读者提高分析问题和解决问题的能力提供了很好的途径。

由于时间仓促及水平有限，书中如有错误和不足之处，恳请广大读者批评指正。

编者

2003.3 于南京理工大学

目 录

实验预备知识	1
一、汇编语言程序设计上机操作过程	1
二、TDEBUG 软件的使用说明	4
软件实验部分	8
第一部分 指令系统	8
实验 1 寻址方式	8
实验 2 标志寄存器	9
实验 3 十进制加法运算指令	11
实验 4 乘法运算指令	11
实验 5 逻辑运算指令	12
实验 6 循环移位指令	12
实验 7 串操作类指令	13
实验 8 程序控制类指令	13
实验 9 80386、80486 增强与增加的指令	14
实验 10 DOS 系统功能调用	15
实验 11 运算符与伪指令系统	18
实验 12 结构、记录和宏	19
第二部分 汇编语言程序设计	21
实验 1 汇编语言程序的调试与运行	23
实验 2 分支程序设计(一)	27
实验 3 分支程序设计(二)	29
实验 4 循环程序设计(一)	30
实验 5 循环程序设计(二)	33
实验 6 子程序结构(一)	34
实验 7 子程序结构(二)	38
实验 8 汇编语言与高级语言接口(一)	39
实验 9 汇编语言与高级语言接口(二)	41
硬件实验部分	47
第一部分 常用芯片应用	47
实验 1 可编程计数器/定时器 8253	47
实验 2 静态存储器 6116	49
实验 3 中断控制器 8259A	52
实验 4 可编程并行接口 8255A	56
实验 4.1 模拟交通信号灯管理	56
实验 4.2 开关量的输入与输出	59

实验 4.3 打印机接口	62
实验 4.4 小键盘输入与主机显示器输出	64
实验 4.5 点阵屏汉字显示	69
实验 5 可编程串行接口 8251A	74
实验 6 可编程 DMA 控制器 8237A	79
实验 7 D/A 转换器	83
实验 8 A/D 转换器	86
第二部分 多芯片综合应用	89
实验 1 综合实验 (一)	89
实验 2 综合实验 (二)	94
实验 3 综合实验 (三)	110
实验 4 综合实验 (四)	116
附录	122
附录一 80X86 指令表及伪指令助记符分类	122
附录二 宏汇编错误信息表	125
附录三 调试程序 DEBUG	129
附录四 TPC-1 型实验箱面板布置图	130
附录五 部分芯片端子	131
参考文献	132

实验预备知识

一、汇编语言程序设计上机操作过程

要建立和运行汇编语言程序，至少要在磁盘上具有如下文件。

编辑程序 (如 TC、EDIT、PE、记事本等)

宏汇编程序 (MASM、ASM)

链接程序 (LINK)

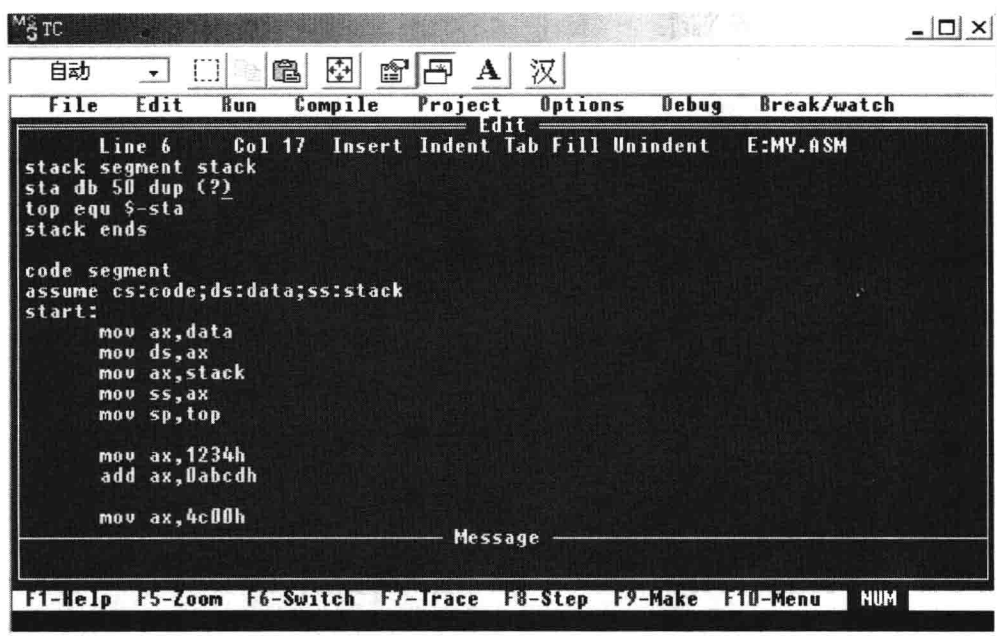
调试程序 (DEBUG、TDEBUG 等)

汇编语言程序上机的四个步骤如下。

步骤一

建立或修改汇编语言源程序(如 MY. ASM 文件)。

用户可通过编辑程序对汇编语言源程序 MY. ASM 进行编写或修改，编辑程序的使用方法用户可查阅相关书籍。



步骤二

用汇编程序 MASM 对源文件进行汇编产生二进制的目标文件(如 MY. OBJ 文件)。

用户可以在源程序建立以后，使用汇编程序 MASM. EXE 对源程序进行汇编。所谓汇编，实际上就是把 (*. ASM) 源程序转换成用二进制代码表示的 (*. OBJ) 目标文件。

在汇编过程中，汇编程序对源程序文件 (*. ASM) 进行二次扫描，如果源程序中有语法错误，则汇编过程结束后，汇编程序会指出源程序中的错误。用户可以再使用编辑程序来修改源程序中的错误，再一次进行汇编，直至得到没有语法错误的 (*. OBJ) 目标文件。

汇编程序的输入文件只有一个，是用户编写的汇编源程序文件 (*.ASM)。

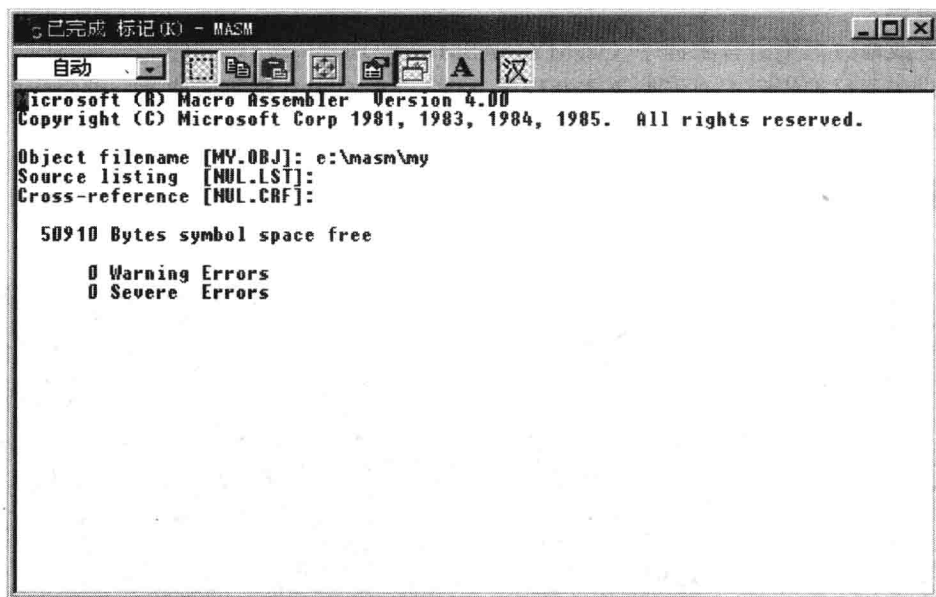
汇编程序的输出文件可以有三个。

① 目标文件 (*.OBJ) 进行汇编的主要目的。

② 列表文件 (*.LST) 可以同时给出源程序和机器语言程序，为调试程序时提供方便，但列表文件是可有可无的。若不需要时，则在屏幕上出现提示信息 [NUL.LST]：时打入回车即可；若需要时，则打入文件名和回车。

③ 交叉引用表 (*.CRF) 同列表文件一样，也是为了调试程序而设置的，交叉文件也是可有可无的。若不需要时，可在屏幕上出现提示信息 [NUL.CRF]：时打入回车即可；若需要时，则打入文件名和回车。

汇编过程结束时，汇编程序给出源程序中的警告性错误 [Warning Error] 和严重错误 [Severe Error]，前者指的是一般性错误而后者指的是语法错误。当存在这两类错误时，屏幕上会指出错误的个数，给出错误信息代号，程序员可以通过查找手册搞清楚错误的性质。要指出的是汇编过程只能指出源程序中的语法错误，并不能指出算法错误和其他错误。



步骤三

用链接程序 LINK 生成可执行文件 (*.EXE 文件)。

汇编程序根据源程序生成二进制的目标文件 (*.OBJ 文件)，但 (*.OBJ 文件)用的是浮动地址，它不能直接上机执行，因此，还必须使用链接程序 LINK 将 (*.OBJ) 文件转换成可执行的文件 (*.EXE 文件)。LINK 还可以将一个目标文件和其他多个模块链接起来。

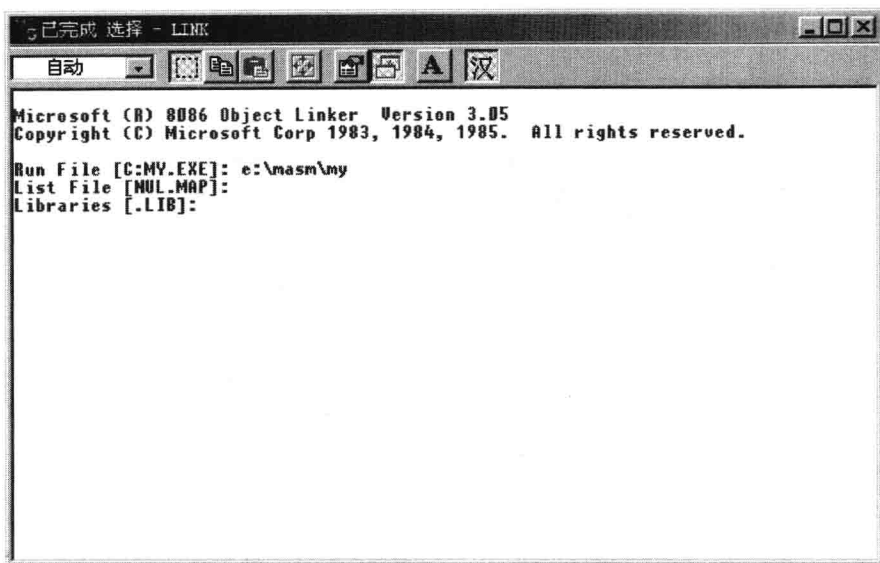
链接程序的输入文件只有一个，是经过汇编产生的目标文件 (*.OBJ)。

链接程序的输出文件可以有两个。

① 可执行文件 (*.EXE 文件) 是 LINK 过程的主要目的。

② 列表分配文件或称映象文件 (*.MAP 文件) 它给出每个段在内存中的分配情况。 (*.MAP) 文件是可有可无的。需要时，打入文件名和回车，否则直接打入回车。

在 LINK 过程的提示信息中，可能会看到最后给出了一个“无堆栈段”的警告性错误，这并不影响程序的执行，如果源程序中设置了堆栈段则无此提示信息。



步骤四

程序的执行。

有了可执行文件，就可以执行程序了，执行程序有两种方法。

方法一

在 DOS 状态下，直接打入文件名后回车，或在 WINDOWS 状态下，双击文件名。

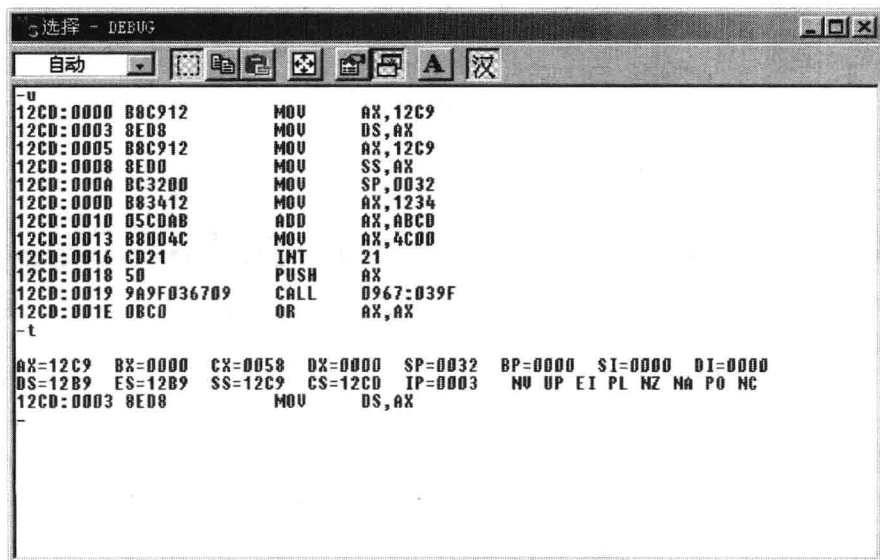
方法二

使用调试程序 TDEBUG 或 DEBUG，操作方法可见 TDEBUG 软件的使用，或参阅附录。

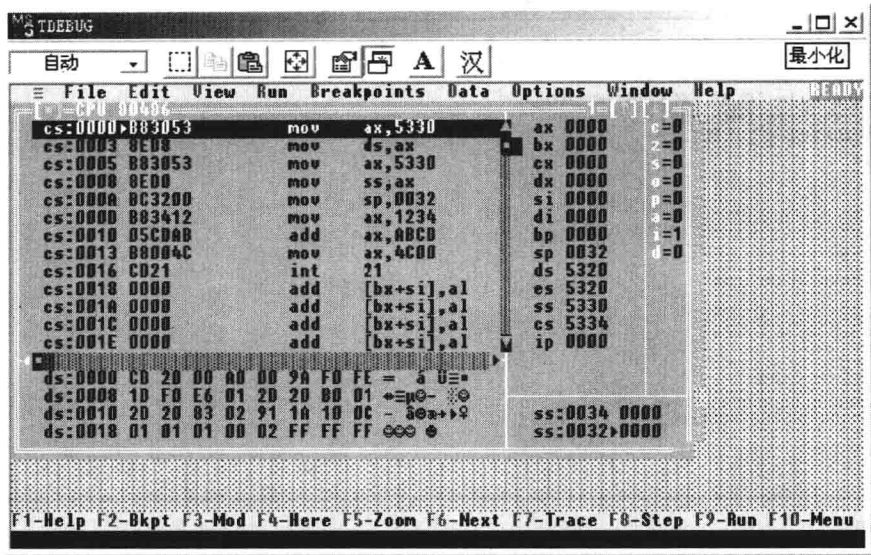
实际上，大部分用户程序都必须经过调试阶段才能纠正程序中的错误，从而得到正确的结果。所谓调试阶段，就是用调试程序（DEBUG）发现错误，再经过编辑、汇编、链接来纠正错误。

关于 DEBUG 程序中的常用命令，可参阅附录。

① 使用 DEBUG 调试如下。

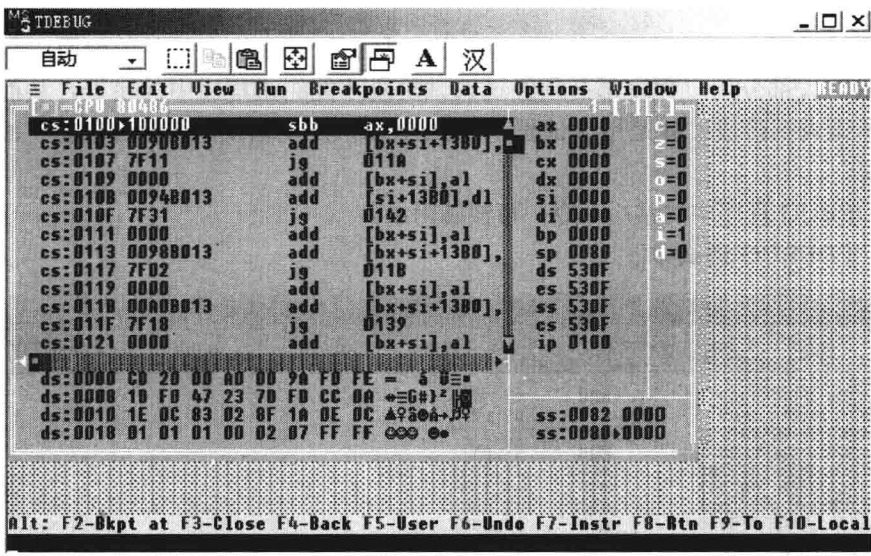


② 使用 TDEBUG 调试如下。



二、TDEBUG 软件的使用说明

Turbo Debugger 3.0 版本的调试环境分为四个窗口：被调试程序窗口、寄存器当前内容窗口、数据段内容窗口和堆栈段内容窗口。如下所示。

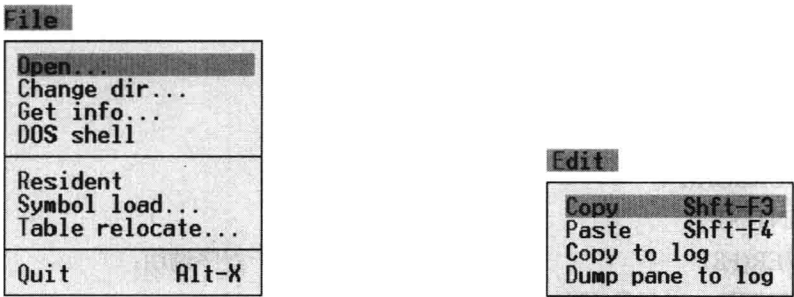


主菜单有：File、Edit、View、Run、Breakpoints、Data、Options、Window、Help 等。下面分别简要介绍。

1. FILE

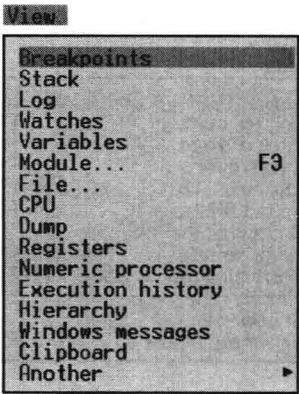
- | | |
|---------------|--------------------|
| OPEN... | 载入一个新的可执行文件； |
| CHANGE DIR... | 改变当前目录； |
| GET INFO... | 显示正在被调试的程序相关的系统信息； |
| DOS SHELL | 回到 DOS 环境； |

RESIDENT	使 TDEBUG 结束调试但不退出调试环境;
SYMBOL LOAD...	载入新的符号表;
TABLE RELOCATE...	给符号表规定一个新的段基址;
QUIT (ALT+X)	从 TDEBUG 中退出。



2. EDIT

COPY (SHIFT+F3)	复制;
PASTE (SHIFT+F4)	粘贴;
COPY TO LOG	将所选项目放到调试运行记录中;
DUMP PANE TO LOG	复制当前窗格到调试运行记录中。



3. VIEW

BREAKPOINT	打开断点窗口;
STACK	打开堆栈窗口;
LOG	打开调试记录窗口;
WATCH	打开观察窗口;
VARIABLE	打开变量窗口;
MODULE... (F3)	打开模块对话框;
FILE...	打开文件对话框;
CPU	打开 CPU 窗口;
DUMP	查看数据段的内容;
REGISTERS	查看寄存器内容;
NUMERIC PROCESSOR	查看数字协处理器 (80X87) 的信息;
EXECUTION HISTORY	查看历史信息;

HIERARCHY	打开类层次结构对话框;
CLIPBOARD	打开剪贴板窗口;
ANOTHE	
MODULE...	打开新的模块化对话框;
DUMP	打开新的数据段窗口;
FILE...	打开另一个文件窗口。

4. RUN

RUN (F9)	运行整个程序;
GO TO CURSOR(F4)	运行到当前位置;
TRACE INTO(F7)	单步运行;
STEP OVER(F8)	单步, 并且跳过子程序调用;
EXECUTE TO... (ALT+F9)	运行到指定位置;
UNTIL RETURN(ALT+F8)	运行子程序, 直到返回到它的调用处;
ANIMATE...	连续单步运行, 对话框中键入两个单步运行间的延时;
BACK TRACE(ALT+F4)	单步的回溯;
INSTRUCTION TRACE(ALT+F7)	执行一条机器指令;
ARGUMENT...	在被调试的程序中设命令行;
PROGRAMME RESET(CTRL+F2)	重新载入程序。

Run

Run	F9
Go to cursor	F4
Trace into	F7
Step over	F8
Execute to...	Alt-F9
Until return	Alt-F8
Animate...	
Back trace	Alt-F4
Instruction trace	Alt-F7
Arguments...	
Program reset	Ctrl-F2

5. BREAKPOINTS

TOGGLE(F2)	在当前位置切换断点;
AT... (ALT+F2)	在指定位置设置断点;
CHANGED MEMORY GLOBAL...	在存储区中设置断点;
XPRESSION TRUE GLOBAL ...	在表达式或变量为真的位置设置断点;
HARDWARE BREAKPOINT ...	设置硬件断点, 用于调试硬件问题;
DELDTE ALL	移去所有断点。

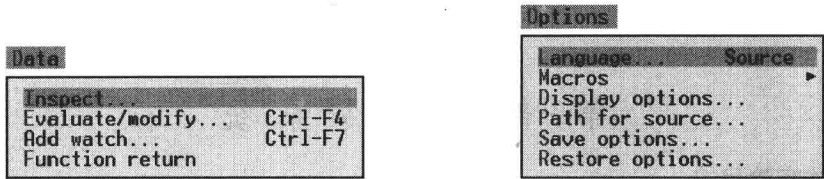
Breakpoints

Toggle	F2
At...	Alt-F2
Changed memory global...	
Expression true global...	
Hardware breakpoint...	
Delete all	

6. DATA

INSPECT...	查找指定的变量或表达式;
------------	--------------

EVALUATE/MODIFY... (CTRL+F4) 计算并显示指定的表达式的值;
ADD WATCH... (CTRL+F7) 在观察窗口添加新的跟踪表达式;
FUNCTION RETURN 显示函数的返回值。



7. OPTIONS

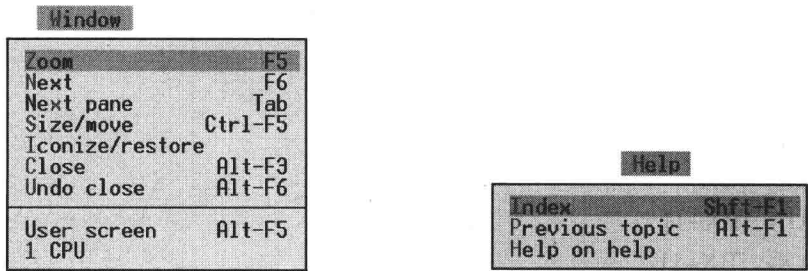
LANGUAGE 设定被调试程序的描述语言;
MACROS

CREATE... ALT=) 设置键击宏指令;
STOP RECORDING(ALT+--) 停止对宏的记录;
REMOVE 移去一个宏;
DELETE ALL 删去所有宏;

DISPLAY OPTIONS... 设置屏幕显示的外观;
PATH FOR SOURCE... 设定搜索源程序的默认路径;
SAVE OPTIONS... 将当前配置保存到磁盘上;
RESTORE OPTIONS... 从磁盘读入以前保存的配置。

8. WINDOW

ZOOM(F5) 缩放当前窗口;
NEXT(F6) 切换到下一个窗口;
NEXT PANE(TAB) 在同一个窗口的各个窗格间进行切换;
SIZE/MOVE(CTRL+F5) 移动窗口或改变窗口的大小;
ICONIZE/RESTORE 窗口最小化/恢复原来的窗口;
CLOSE(ALT+F3) 关闭当前窗口;
UNDO CLOSE(ALT+F6) 恢复最近被关闭的窗口;
USER SCREEN(ALT+F5) 查看被调试程序反映在屏幕上的运行结果;
1 CPU 切换到高亮窗口。



9. HELP

INDEX(SHIFT+F1) 打开联机帮助目录;
PREVIOUS TOPIC(ALT+F1) 回到前一个帮助主题;
HELP ON HELP 关于如何使用联机帮助的帮助信息。

软件实验部分

第一部分 指令系统

一、总体实验目的

- ① 学习掌握调试软件 (DEBUG 或 TDEBUG) 的基本使用方法。
- ② 掌握 8086/8088 的寻址方式。
- ③ 了解状态标志与控制标志的特点。
- ④ 熟练掌握指令系统的基本操作。
- ⑤ 掌握 DOS 系统功能调用。
- ⑥ 掌握运算符及伪指令系统的基本操作。

二、总体实验内容

将寻址方式、状态标志、指令系统及运算符与伪指令系统等内容分为几部分, 对其分别进行调试与验证。

实验1 寻址方式

已知: (60000H)=12H, (60001H)=34H, (60002H)=56H, (60003H)=78H, (70000H)=0ABH, (70001H)=0CDH, (70002H)=0EFH, (70003H)=0DH。

- ① 分别执行下列指令后, 填入指定寄存器的当前内容。

MOV AX, 6000H	
MOV DS, AX	
MOV AX, 7000H	
MOV SS, AX	
MOV BX, 0	
MOV BP, 0	
MOV SI, 2	
MOV AX, BX	AX=_____
MOV AX, [0000H]	AX=_____
MOV AL, [0000H]	AL=_____
MOV AX, [BX]	AX=_____
MOV AL, [BX+1]	AL=_____
MOV AX, [BX+1]	AX=_____
MOV AL, [BX+2]	AL=_____
MOV AX, [BX+2]	AX=_____
MOV AX, [BX+SI]	AX=_____
MOV AX, [BP+SI]	AX=_____

② 按下列要求分别编写指令组，将 BLOCK 数组中的第六个字数据存放在 DX 寄存器中。

```
BLOCK DW 1234H, 5678H, 'AB', 'CD'
寄存器间接寻址
```

寄存器相对寻址

基址加变址寻址

实验2 标志寄存器

① MOV AL, 76H

ADD AL, 0A6H

将操作码改为 ADC、SUB、SBB，分别执行后填表。

操 作 码	AL	OF	SF	ZF	AF	PF	CF
ADD							
ADC							
SUB							
SBB							

② MOV AX, 2376H

ADD AX, 00A6H

将操作码改为 ADC、SUB、SBB，分别执行后填表。

操 作 码	AX	OF	SF	ZF	AF	PF	CF
ADD							
ADC							
SUB							
SBB							

③ MOV AX, 0A95BH

ADD AX, 8CA2H

将操作码改为 ADC、SUB、SBB，分别执行后填表。

操 作 码	AX	OF	SF	ZF	AF	PF	CF
ADD							
ADC							
SUB							
SBB							

④ MOV AX, 8889H

ADD AX, 7777H

将操作码改为 ADC、SUB、SBB，分别执行后填表。

操 作 码	AX	OF	SF	ZF	AF	PF	CF
ADD							
ADC							
SUB							
SBB							

⑤ MOV AX, 8765H

AND AX, 89ABH

将操作码改为 OR、XOR、TEST，分别执行后填表。

操 作 码	AX	OF	SF	ZF	PF	CF
AND						
OR						
XOR						
TEST						

⑥ MOV AX, 2376H

SHL AX, 1

将操作码改为 SHR、SAR，分别执行后填表。

操 作 码	AX	OF	SF	ZF	PF	CF
SHL						
SHR						
SAR						

⑦ MOV AX, 2376H

MOV CL, 4

SHL AX, CL

将操作码改为 SHR、SAR，分别执行后填表。

操 作 码	AX	OF	SF	ZF	PF	CF
SHL						
SHR						
SAR						

⑧ CLC

MOV AX, 0A95BH

ROL AX, 1

将操作码改为 ROR、RCL、RCR，分别执行后填表。

操 作 码	AX	OF	CF
ROL			
ROR			
RCL			
RCR			

```

⑨ STC
MOV AX, 0A95BH
ROL AX, 1

```

将操作码改为 ROR、RCL、RCR，分别执行后填表。

操 作 码	AX	OF	CF
ROL			
ROR			
RCL			
RCR			

实验 3 十进制加法运算指令

```

① MOV AL, 78H
   ADD AL, 26H      AL=_____
   DAA              AL=_____ CF=_____ AF=_____

② MOV AX, 78H
   ADD AX, 26H      AX=_____
   DAA              AX=_____ CF=_____ AF=_____

③ MOV AX, 5678H
   ADD AX, 1224H     AX=_____
   DAA              AX=_____ CF=_____ AF=_____

④ MOV AL, 07H
   ADD AL, 09H      AX=_____
   DAA              AX=_____ CF=_____ AF=_____

⑤ MOV AL, 07H
   ADD AL, 09H      AX=_____
   AAA              AX=_____ CF=_____ AF=_____

```

试将上述指令改为减法指令后，再填入指定的寄存器或标志位的值。

实验 4 乘法运算指令

```

① MOV AL, 96H

```