

实验 4 顺序结构程序设计

一、实验目的

熟悉汇编语言的上机环境，学习 EDIT,MASM,LINK 程序的使用，掌握建立、汇编、链接、运行汇编语言程序的过程；掌握常用指令的用法；掌握完整的汇编程序编写过程；掌握汇编程序调试过程。

二、程序阅读题

试问如下程序执行到 exit 时，字节单元 a、b、cc、cc+1、cc+2 中的内容各是什么？源程序如下：

```
DATA SEGMENT
    A DB 0
    B DB 0
    CC DB 30H,40H,50H
DATA ENDS
CODE SEGMENT
    ASSUME CS:CODE,DS:DATA
START:
    MOV AX, DATA
    MOV DS, AX
    MOV AL,CC+1
    ADD AL,CC
    MOV A,AL
    MOV AL,CC+2
    SUB AL,CC+1
    MOV B,AL
    ADD CC,10H
    ADD CC+1,20H
    MOV AL, A
    ADD CC+2,AL
EXIT:  MOV AH,4CH
        INT 21H
CODE ENDS
        END START
```

```

-t
AX=0770 BX=0000 CX=003E DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0026  NU UP EI PL NZ NA PE NC
0771:0026 00060400      ADD     [0004],AL      DS:0004=50
-t
AX=0770 BX=0000 CX=003E DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=002A  OU UP EI NG NZ NA PE NC
0771:002A B44C      MOV     AH,4C
-t
AX=4C70 BX=0000 CX=003E DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=002C  OU UP EI NG NZ NA PE NC
0771:002C CD21      INT     21
-d ds:0
0770:0000 70 10 40 60 C0 00 00 00-00 00 00 00 00 00 00 00 00  p.e`.
0770:0010 B8 70 07 8E D8 A0 03 00-02 06 02 00 A2 00 00 A0 00  .p.
0770:0020 04 00 2A 06 03 00 A2 01-00 80 06 02 00 10 80 06  ..*.
0770:0030 03 00 20 A0 00 00 00 00-04 00 B4 4C CD 21 00 00  ..L.F.
0770:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
0770:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
0770:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
0770:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .....
;

```

当程序执行到 exit 时，字节单元

- A = 70H
- B = 10H
- CC = 40H
- CC+1 = 60H
- CC+2 = C0H

阅读以上程序，给程序添加注释，分析出结果，并上机验证。

三、实验题

1. 在 ARRAY 数组中依次存储了七个字数据，紧接着是名为 ZERO 的字单元，表示如下：

ARRAY DW 23, 36, 2, 100, 32000, 54, 11

ZERO DW ?

```

; 定义数据段
DATA SEGMENT
    ARRAY DW 23, 36, 2, 100, 32000, 54, 11 ; 七个字数据的数组
    ZERO DW ? ; 未初始化的字单元
DATA ENDS

; 定义代码段
CODE SEGMENT
    ASSUME CS:CODE, DS:DATA

START:
    ; 初始化数据段寄存器
    MOV AX, DATA
    MOV DS, AX

    ; (1) 如果 BX 包含数组 ARRAY 的初始地址
    LEA BX, ARRAY ; 将 ARRAY 的地址加载到 BX
    MOV AX, [BX+12] ; 获取 ARRAY 中偏移量为 12 的数据(即 11)
    MOV ZERO, AX ; 将 11 存储到 ZERO 单元

    ; 可以添加一些指令来查看结果
    MOV AX, ZERO ; 将 ZERO 的值加载到 AX

    ; (2) 如果 BX 包含数据 11 在数组中的偏移量
    MOV BX, 12 ; 11 在数组中的偏移量是 12(6*2=12, 因为每个 DW 占 2
字节)
    MOV AX, ARRAY[BX] ; 获取 ARRAY 中偏移量为 BX 的数据(即 11)
    MOV ZERO, AX ; 将 11 存储到 ZERO 单元

    ; 程序退出
    MOV AH, 4CH
    INT 21H
CODE ENDS
END START

```

- (1) 如果 BX 包含数组 ARRAY 的初始地址，请编写指令将数据 11 传送给 ZERO 单元。使用 debug 命令查看运行结果，截图放在下面。

Mov AX [BX+12]

```

AX=0770 BX=0000 CX=0030 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0760 ES=0760 SS=076F CS=0771 IP=0003  NU UP EI PL NZ NA PO NC
0771:0003 8ED8      MOV     DS,AX
-t
AX=0770 BX=0000 CX=0030 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0005  NU UP EI PL NZ NA PO NC
0771:0005 BD1E0000  LEA     BX,[0000]      DS:0000=0017
-t
AX=0770 BX=0000 CX=0030 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0009  NU UP EI PL NZ NA PO NC
0771:0009 BB470C      MOV     AX,[BX+0C]      DS:000C=000B
-t
AX=000B BX=0000 CX=0030 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=000C  NU UP EI PL NZ NA PO NC
0771:000C A30E00      MOV     [000E],AX      DS:000E=0000
-t
AX=000B BX=0000 CX=0030 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=000F  NU UP EI PL NZ NA PO NC
0771:000F A10E00      MOV     AX,[000E]      DS:000E=000B
-t

```

(2) 如果 BX 包含数据 11 在数组中的位移量,请编写指令将数据 11 传送给 ZERO 单元。使用 debug 命令查看运行结果,截图放在下面。Zero 设置的是[000E]

MOV BX, 12

MOV AX, ARRAY[BX]

```

DS=0770 ES=0760 SS=076F CS=0771 IP=000F  NU UP EI PL NZ NA PO NC
0771:000F A10E00      MOV     AX,[000E]      DS:000E=000B
-t
AX=000B BX=0000 CX=0030 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0012  NU UP EI PL NZ NA PO NC
0771:0012 BB0C00      MOV     BX,000C
-t
AX=000B BX=000C CX=0030 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0015  NU UP EI PL NZ NA PO NC
0771:0015 BB870000  MOV     AX,[BX+0000]   DS:000C=000B
-t
AX=000B BX=000C CX=0030 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0019  NU UP EI PL NZ NA PO NC
0771:0019 A30E00      MOV     [000E],AX      DS:000E=000B
-t

```

2. 设 x, y, z 均为双精度数, 它们分别存放在地址为 X, X+2; Y, Y+2; Z, Z+2 的存储单元中, 存放时高位字在高地址中, 低位字在低地址中。用指令序列实现 $X+Y+24-Z$, 并把结果存入 W 单元。把源程序附在下面, 并把 debug 单步调试的截图附在下面。(备注: x,y,z 的值自行给)

```

DATA SEGMENT
    X DW 1234H, 5678H ; 双精度数 x (低位在前, 高位在后)
    Y DW 2345H, 6789H ; 双精度数 y (低位在前, 高位在后)
    Z DW 1111H, 2222H ; 双精度数 z (低位在前, 高位在后)
    W DW 0, 0 ; 结果存放位置 (低位在前, 高位在后)
    CONST_24 DW 24, 0 ; 常数 24 (双精度)
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA
START:
    MOV AX, DATA
    MOV DS, AX

```

```

; 计算 X + Y
MOV AX, X          ; 加载 X 的低位
ADD AX, Y           ; 加上 Y 的低位
MOV BX, AX          ; 保存低位结果

MOV AX, X+2         ; 加载 X 的高位
ADC AX, Y+2         ; 带进位加上 Y 的高位
MOV DX, AX          ; 保存高位结果

; 计算 (X + Y) + 24
MOV AX, BX          ; 恢复低位结果
ADD AX, CONST_24    ; 加上 24 的低位
MOV BX, AX          ; 保存低位结果

MOV AX, DX          ; 恢复高位结果
ADC AX, CONST_24+2  ; 带进位加上 24 的高位
MOV DX, AX          ; 保存高位结果

; 计算 (X + Y + 24) - Z
MOV AX, BX          ; 恢复低位结果
SUB AX, Z            ; 减去 Z 的低位
MOV W, AX           ; 保存结果的低位

MOV AX, DX          ; 恢复高位结果
SBB AX, Z+2         ; 带借位减去 Z 的高位
MOV W+2, AX         ; 保存结果的高位

MOV AH, 4CH
INT 21H

```

CODE ENDS

END START

```

0772:0003 BFD8      MOV     DS,AX
0772:0005 A10000     MOV     AX,[0000]
0772:0008 03060400   ADD     AX,[0004]
0772:000C BBD8      MOV     BX,AX
0772:000E A10200     MOV     AX,[0002]
0772:0011 13060600   ADC     AX,[0006]
0772:0015 BBD0      MOV     DX,AX
0772:0017 BBC3      MOV     AX,BX
0772:0019 03061000   ADD     AX,[0010]
0772:001D BBD8      MOV     BX,AX
0772:001F BBC2      MOV     AX,DX
-      u
0772:0021 13061200   ADC     AX,[0012]
0772:0025 BBD0      MOV     DX,AX
0772:0027 BBC3      MOV     AX,BX
0772:0029 2B060800   SUB     AX,[0008]
0772:002D A30C00     MOV     [000C],AX
0772:0030 BBC2      MOV     AX,DX
0772:0032 1B060A00   SBB     AX,[000A]
0772:0036 A30E00     MOV     [000E],AX
0772:0039 B44C      MOV     AH,4C
0772:003B CD21      INT     21
0772:003D 0000      ADD     [BX+SI],AL
0772:003F 0000      ADD     [BX+SI],AL
-      ;

```

运行结果:

```
-d DS:c
3770:0000 18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 24 DF 9B
3770:0010 18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3770:0020 BB 70 07 8E D8 A1 00 00 03 06 04 00 8B D8 A1 02 00 00 00 00
3770:0030 00 13 06 06 00 8B D0 8B C3 03 06 10 00 8B D8 8B 00 00 00 00
3770:0040 C2 13 06 12 00 8B D0 8B C3 2B 06 08 00 A3 0C 00 00 00 00 00
3770:0050 8B C2 1B 06 0A 00 A3 0E 00 B4 4C CD 21 00 00 00 00 00 00 00
3770:0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3770:0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3770:0080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

X DW 1234H, 5678H

Y DW 2345H, 6789H

Z DW 1111H, 2222H

W->2480H,9BDFH,结果正确

3. 编写程序, x, y, z, v 均为 16 位带符号数, 计算 $(v - (x*y + z - 540)) / x$ 。把源程序附在下面, 并把 debug 单步调试的截图附在下面。(备注: x, y, z, v 的值自行给)

```
; 定义数据段
DATA SEGMENT
    X DW 5          ; 16 位带符号数 x
    Y DW 10         ; 16 位带符号数 y
    Z DW 100        ; 16 位带符号数 z
    V DW 1000       ; 16 位带符号数 v
    CONST_540 DW 540 ; 常数 540
    RESULT DW ?     ; 存放计算结果
DATA ENDS

; 定义代码段
CODE SEGMENT
    ASSUME CS:CODE, DS:DATA

START:
    ; 初始化数据段寄存器
    MOV AX, DATA
    MOV DS, AX

    ; 计算 x*y
    MOV AX, X      ; 将 x 加载到 AX
    IMUL Y         ; 有符号乘法, 结果在 DX:AX

    ; 计算 x*y + z
    ADD AX, Z      ; 将 z 加到结果的低 16 位
    ADC DX, 0      ; 处理可能的进位
```

```

; 计算  $x*y + z - 540$ 
SUB AX, CONST_540 ; 从结果中减去 540
SBB DX, 0 ; 处理可能的借位

; 保存  $x*y + z - 540$  的结果到 BX:CX
MOV BX, DX
MOV CX, AX

; 计算  $v - (x*y + z - 540)$ 
MOV AX, V ; 将 v 加载到 AX
SUB AX, CX ; 减去之前计算的结果的低 16 位
MOV CX, AX ; 保存结果的低 16 位到 CX

MOV AX, 0 ; 高 16 位设为 0 (因为 v 是 16 位数)
SBB AX, BX ; 减去之前计算的结果的高 16 位, 考虑借位
MOV BX, AX ; 保存结果的高 16 位到 BX

; 计算  $(v - (x*y + z - 540)) / x$ 
; 准备被除数 BX:CX
MOV AX, CX ; 将低 16 位移到 AX
MOV DX, BX ; 将高 16 位移到 DX

; 执行有符号除法
IDIV X ; 有符号除法, 结果在 AX, 余数在 DX

; 保存结果
MOV RESULT, AX

; 程序退出
MOV AH, 4CH
INT 21H
CODE ENDS
END START

```

RESULT 为 0770:000A

```

AX=0116 BX=0000 CX=056E DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0034 NU UP EI PL ZR AC PE CY
0771:0034 A30A00 MOV [000A],AX DS:000A=0000
-t
AX=0116 BX=0000 CX=056E DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0037 NU UP EI PL ZR AC PE CY
0771:0037 B44C MOV AH,4C
-t
AX=4C16 BX=0000 CX=056E DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=076F CS=0771 IP=0039 NU UP EI PL ZR AC PE CY
0771:0039 CD21 INT 21
-d ds:a
0770:0000 16 01 00 00 00 00
0770:0010 B8 70 07 8E DB A1 00 00-F7 2E 02 00 03 06 04 00 .p.....
0770:0020 83 D2 00 2B 06 03 00 83-DA 00 8B DA 8B CB A1 06 ...+.
0770:0030 00 2B C1 8B CB 8B 00 00-1B C3 8B D8 8B C1 8B D3 .>.....L.!.
0770:0040 F7 3E 00 00 A3 0A 00 B4-4C CD Z1 00 00 00 00 00 .>.....L.!.
0770:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
0770:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
0770:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
0770:0080 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....

```

RESULT = 278 = 0116H

三 实验报告

要求： 1.每步操作过程及所用指令；2.用截图方式反应指令执行前后的相关寄存器或存储单元内容的变化，并对所得结果进行分析。