

# 暨南大学本科实验报告专用纸

课程名称 汇编语言实验 题目 bx 和 loop 实验项目编号 5 实验项目类型 上机 实验地点 517 成绩评定 \_\_\_\_\_  
学生姓名 陈文笛 学号 2021103285 学院 网络空间安全 系 网络空间安全 专业 网络空间安全 实验时间 2023 年 3 月 20 日 下 午 温度    °C 湿度     
指导教师 张银炎

## 任务 1

### 实验简介

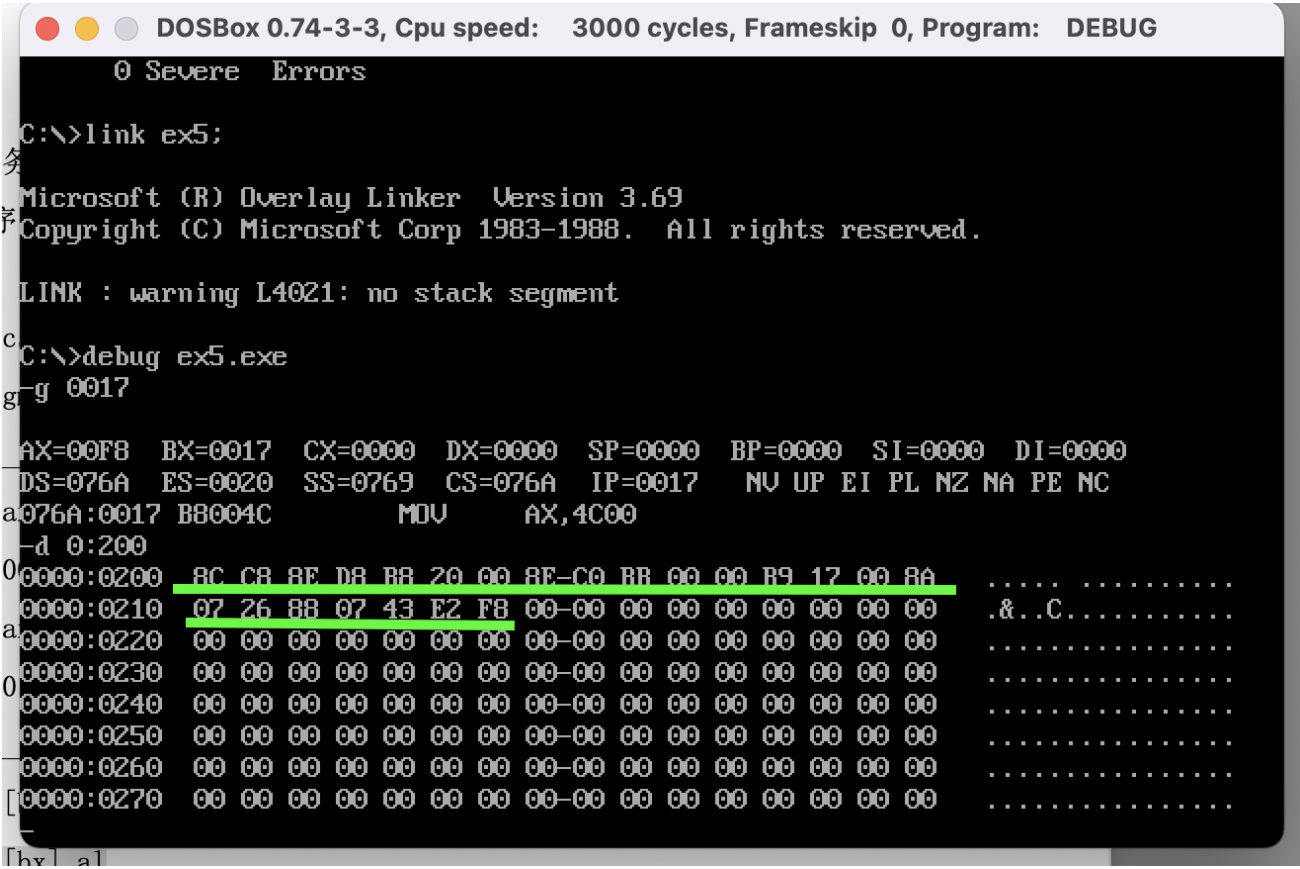
下面程序的功能是将“mov ax,4c00h”之前的指令复制到内存 0:200 处，补全程序。

### 实验结果截图

```
assume cs:code
code segment
mov ax, CS
mov ds, ax
mov ax, 0020h
mov es, ax
mov bx, 0
mov cx, 0017h
s: mov al, [bx]
mov es:[bx], al
inc bx
loop s
mov ax, 4c00h
int 21h
code ends
end
```

1. CS      2. 0017h

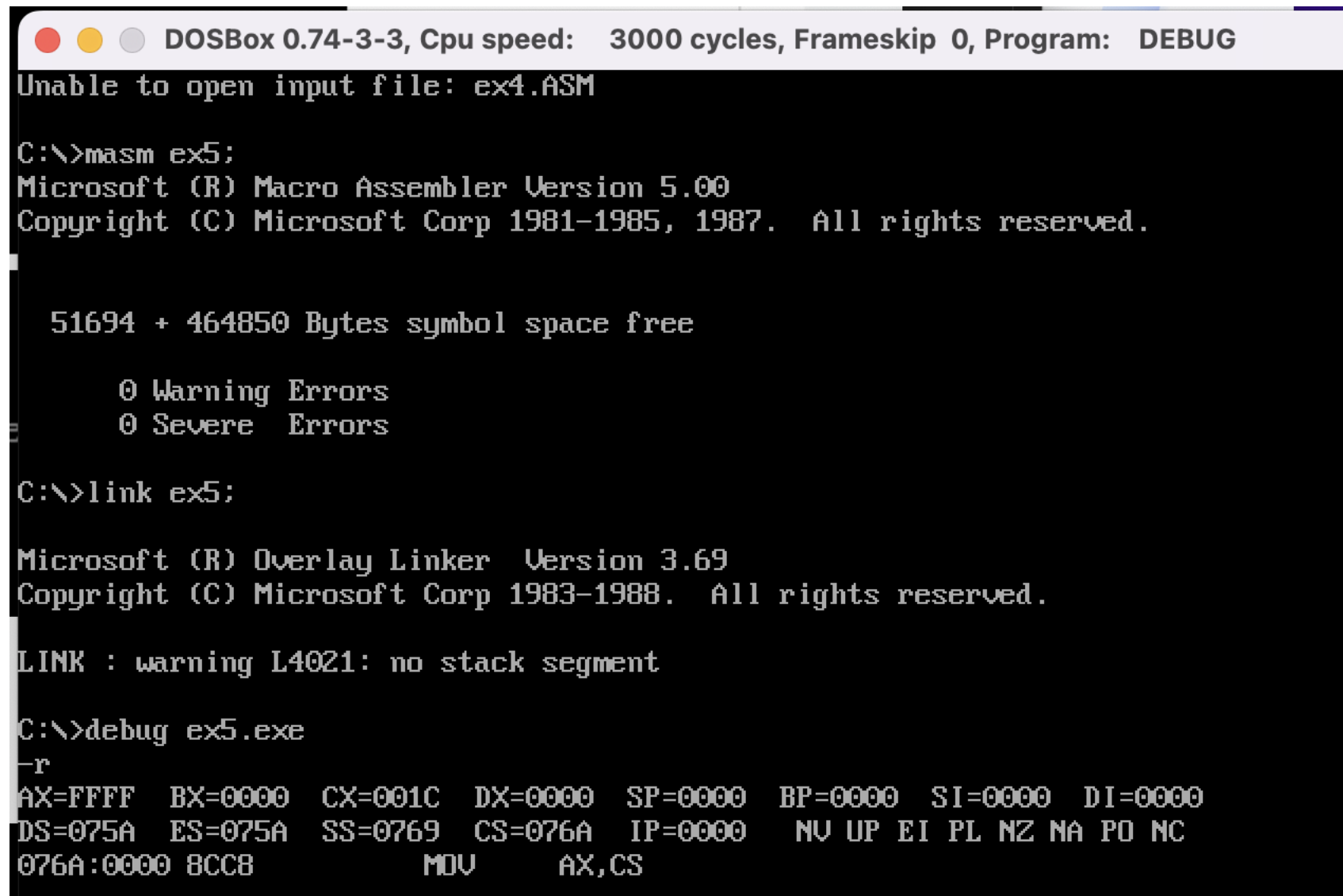
学生签名 陈文笛



## 实验结果分析

1. 空：CS 是代码段的首址寄存器。所以需要复制的指令从 CS 寄存器指示的位置开始。故将 CS 的值通过 ax 赋值给 ds。
2. 空：首先，先随便对 cx 赋一个值 `mov cx, 005h`。然后把程序进行编译链接，再进入调试。

由下图中可见，`cx=001c`，这说明程序的大小是 1c 个字节。



```
DOSBox 0.74-3-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
Unable to open input file: ex4.ASM

C:\>masm ex5;
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981-1985, 1987. All rights reserved.

51694 + 464850 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\>link ex5;

Microsoft (R) Overlay Linker Version 3.69
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

LINK : warning L4021: no stack segment

C:\>debug ex5.exe
-r
AX=FFFF BX=0000 CX=001C DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=075A ES=075A SS=0769 CS=076A IP=0000  NV UP EI PL NZ NA PO NC
076A:0000 8CC8          MOV     AX,CS
```

在用 U 命令将内存中的机器指令翻译成汇编指令。可见，最后

```
mov ax, 4c00h
```

```
int 21h
```

两条命令的总字节数为 5。

所以“`mov ax, 4c00h`”之前的命令字节数为  $1c - 5 = 17$  (字节)

```

-u
076A:0000 8CC8      MOV     AX,CS
076A:0002 8ED8      MOV     DS,AX
076A:0004 B82000     MOV     AX,0020
076A:0007 8EC0      MOV     ES,AX
076A:0009 BB0000     MOV     BX,0000
076A:000C B90500     MOV     CX,0005
076A:000F 8A07      MOV     AL,[BX]
076A:0011 26        ES:
076A:0012 8807      MOV     [BX],AL
076A:0014 43        INC     BX
076A:0015 E2F8      LOOP    000F
076A:0017 B8004C     MOV     AX,4C00
076A:001A CD21      INT     21
076A:001C 80E401     AND     AH,01
076A:001F 8946EC     MOV     [BP-14],AX

```

由于 `mov es:[bx],al` 采用 `al` 进行移动，一次移动 1 字节，所以一个需要移动 17 次。

综上所述，第 2 空填入 0017h

更改程序并再次执行

```

DOSBox 0.74-3-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
0 Severe Errors

C:\>link ex5;

Microsoft (R) Overlay Linker Version 3.69
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

LINK : warning L4021: no stack segment

C:\>debug ex5.exe
-g 0017

AX=00F8 BX=0017 CX=0000 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=076A ES=0020 SS=0769 CS=076A IP=0017  NV UP EI PL NZ NA PE NC
a:076A:0017 B8004C      MOV     AX,4C00
-d 0:200
0:0000:0200  8C CB 8E D8 B8 20 00 8E C0 B8 00 00 B9 17 00 BA .....
a:0000:0210  07 26 88 07 43 E2 F8 00 00 00 00 00 00 00 00 .&..C.....
0:0000:0220  00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0:0000:0230  00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0:0000:0240  00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0:0000:0250  00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0:0000:0260  00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
[0000:0270  00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
[bx] al

```

可见，“`mov ax,4c00h`”之前的指令成功复制到内存 0:200 处

## 任务 2

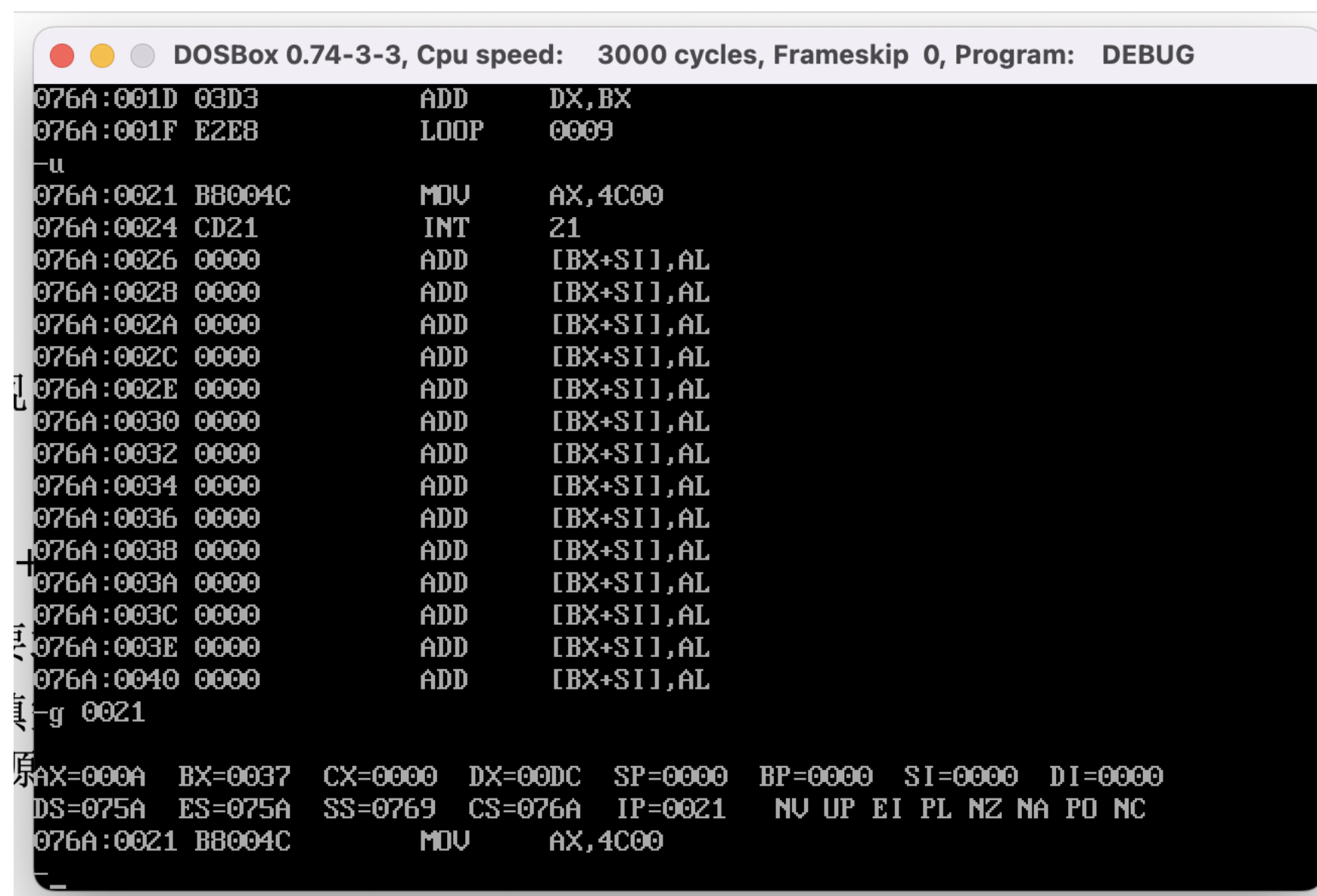
### 实验简介

2. 用 loop 循环实现以下公式的结果计算，并将计算结果保存到 dx 寄存器。

$$y = x_1 + x_2 + \cdots + x_{10}$$

其中  $x_i = 0 + 1 + \cdots + i$  (例如:  $x_5 = 0 + 1 + 2 + 3 + 4 + 5$ )。

### 实验结果截图&实验结果分析



The screenshot shows the DOSBox 0.74-3-3 interface with the following assembly code and register values:

```
DOSBox 0.74-3-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
076A:001D 03D3      ADD     DX,BX
076A:001F E2E8      LOOP   0009
-u
076A:0021 BB004C      MOV     AX,4C00
076A:0024 CD21      INT     21
076A:0026 0000      ADD     [BX+SI],AL
076A:0028 0000      ADD     [BX+SI],AL
076A:002A 0000      ADD     [BX+SI],AL
076A:002C 0000      ADD     [BX+SI],AL
076A:002E 0000      ADD     [BX+SI],AL
076A:0030 0000      ADD     [BX+SI],AL
076A:0032 0000      ADD     [BX+SI],AL
076A:0034 0000      ADD     [BX+SI],AL
076A:0036 0000      ADD     [BX+SI],AL
076A:0038 0000      ADD     [BX+SI],AL
076A:003A 0000      ADD     [BX+SI],AL
076A:003C 0000      ADD     [BX+SI],AL
076A:003E 0000      ADD     [BX+SI],AL
076A:0040 0000      ADD     [BX+SI],AL
-g 0021
AX=000A  BX=0037  CX=0000  DX=00DC  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076A  IP=0021  NU UP EI PL NZ NA PO NC
076A:0021 BB004C      MOV     AX,4C00
```

可见 DX=00DC, 00DC (h)=220 (D) 与计算结果一致

## 遇到的问题及解决方法

一开始我漏了 `pop cx`，所以执行的时候-g 一直无法结束。这是由于出来循环 S2 时，`cx=0`,如果这时不 `pop cx`，执行完第一个 S1 之后，`cx=cx-1`,此时 `cx=FFFFh`，将进入无限循环。

## 代码

```
assume cs:code

code segment

    mov dx,0

    mov ax,0

    mov cx,10
s1: add ax,1

    push cx

    mov bl,0

    mov bh,0

    mov cx,ax
s2: add bh,1

    add bl,bh
```

```
loop s2  
  
pop cx  
  
mov bh,0  
  
mov dx,bx  
  
loop s1  
  
mov ax,4c00h  
  
int 21h
```

```
code ends
```

```
end
```

