**实验一 汇编语言程序的上机过程**

班级： 22计科3班 姓名： 张雅瑞 学号： 2022334323029

**一、实验目的**

* 1. 掌握汇编语言程序的上机步骤

(1 )用编辑程序(如EDIT、记事本等)编辑汇编语言源程序(建立.ASM文件)

(2 )用MASM程序产生OBJ文件

(3 )用LINK程序产生EXE文件

(4 )程序的运行(用DEBUG或在DOS下直接运行)

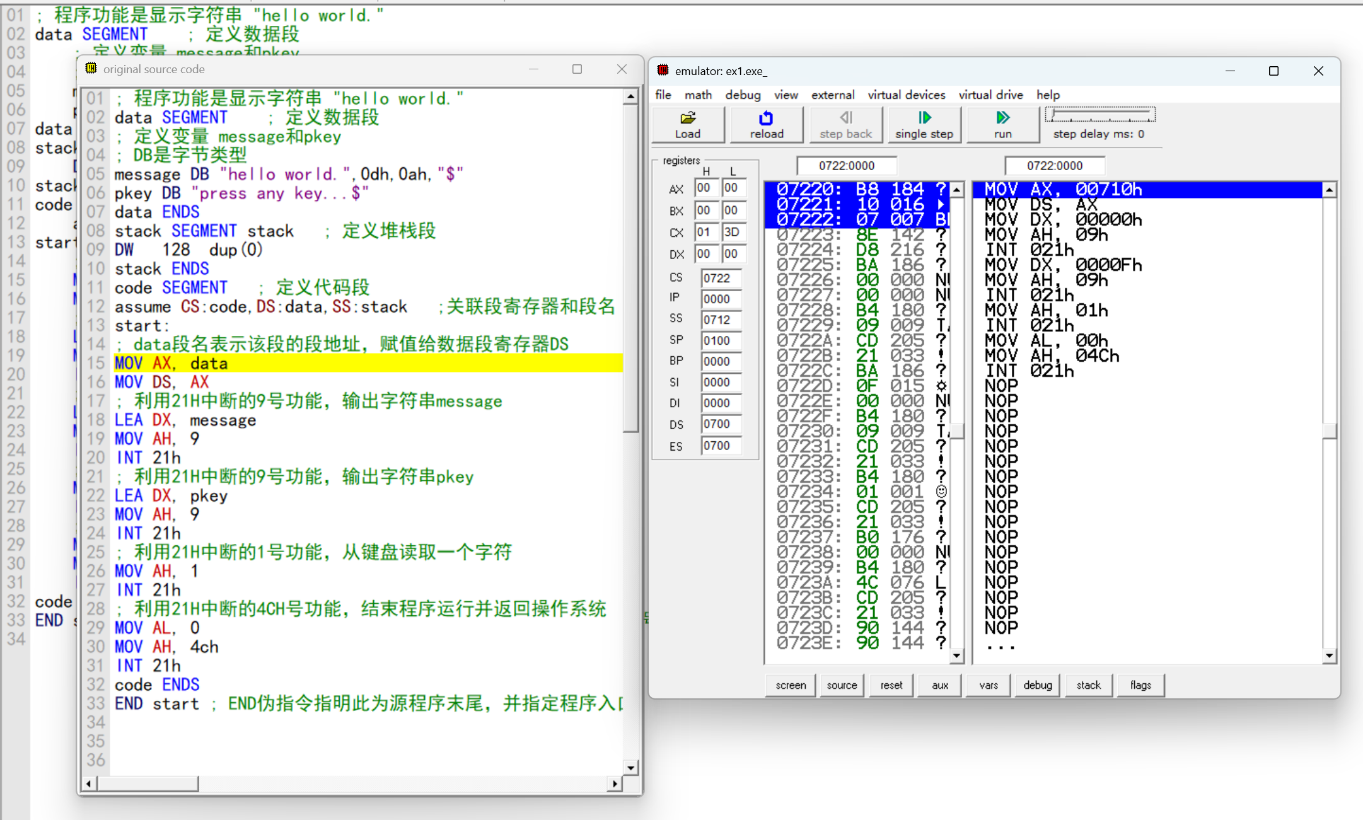
* 1. 掌握程序的调试方法

**二、实验内容**

本次实验你所完成的具体内容加以说明。

任务1：

1. EMU8086调试

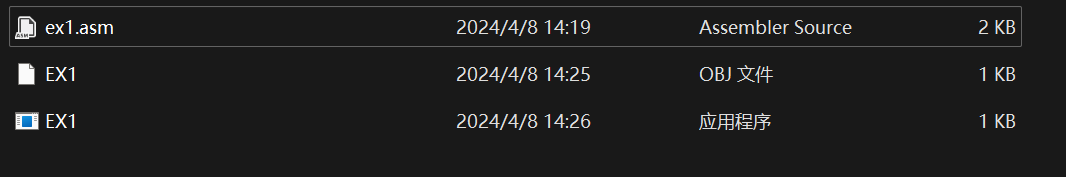


代码执行

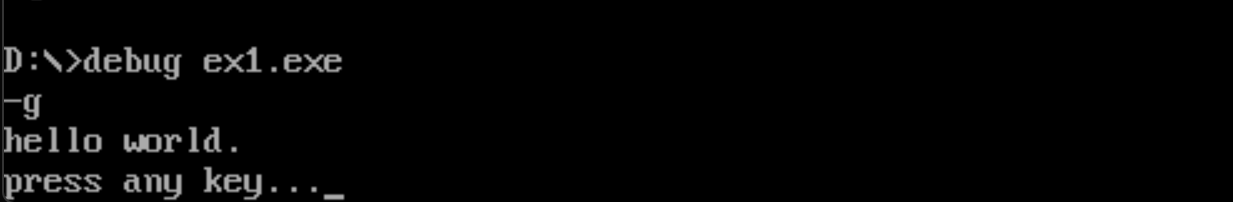


调试结果

2）MASM调试

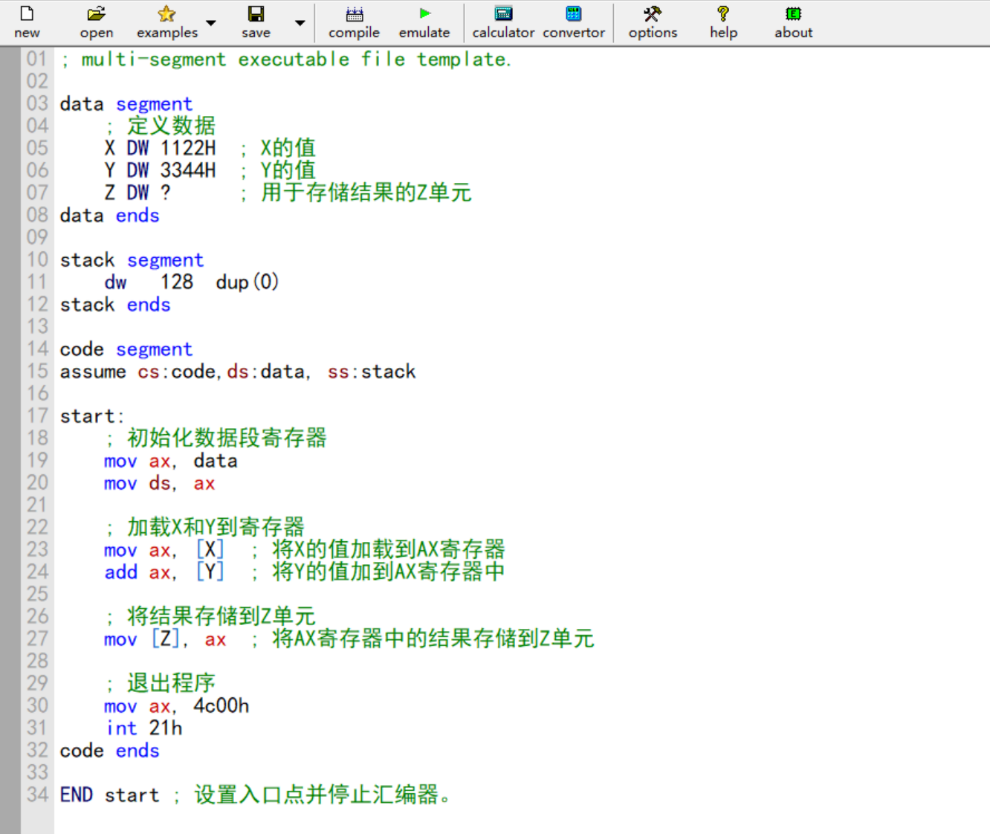


用masm、link指令生成文件

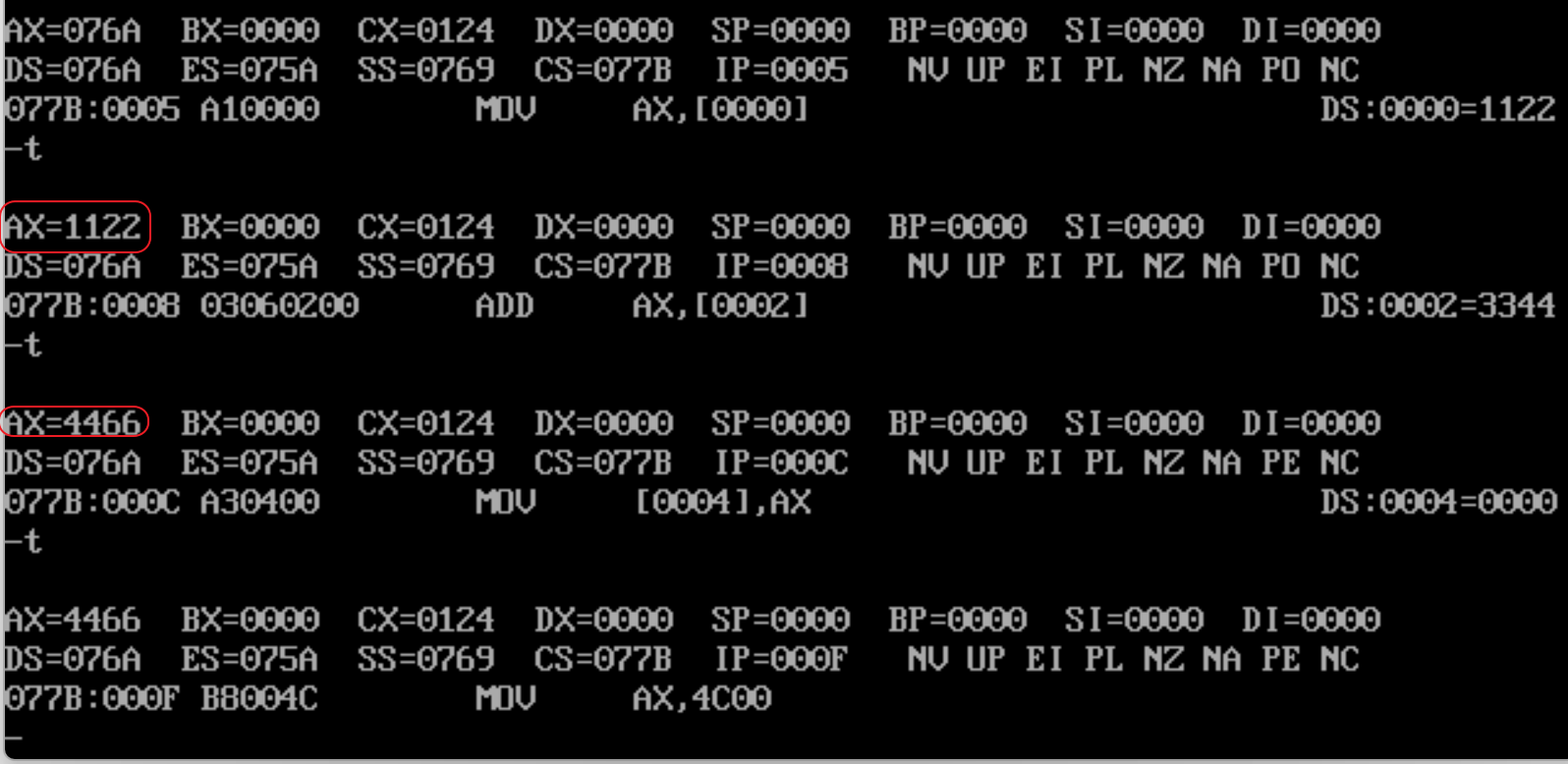


Debug调试成功

任务2：

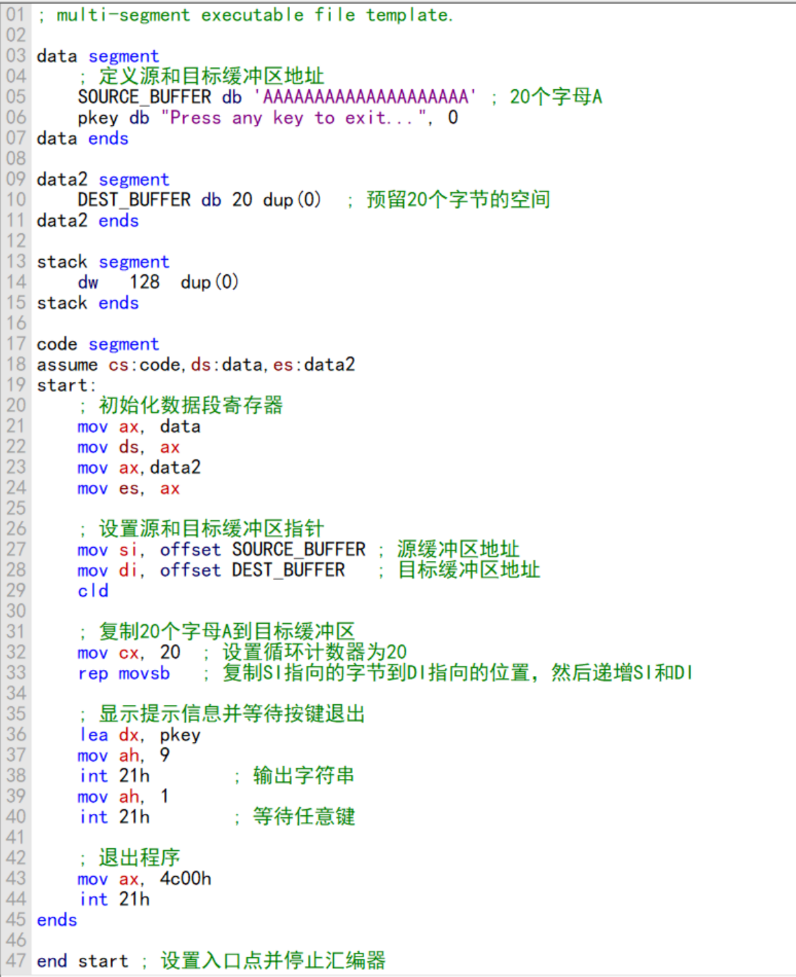


代码

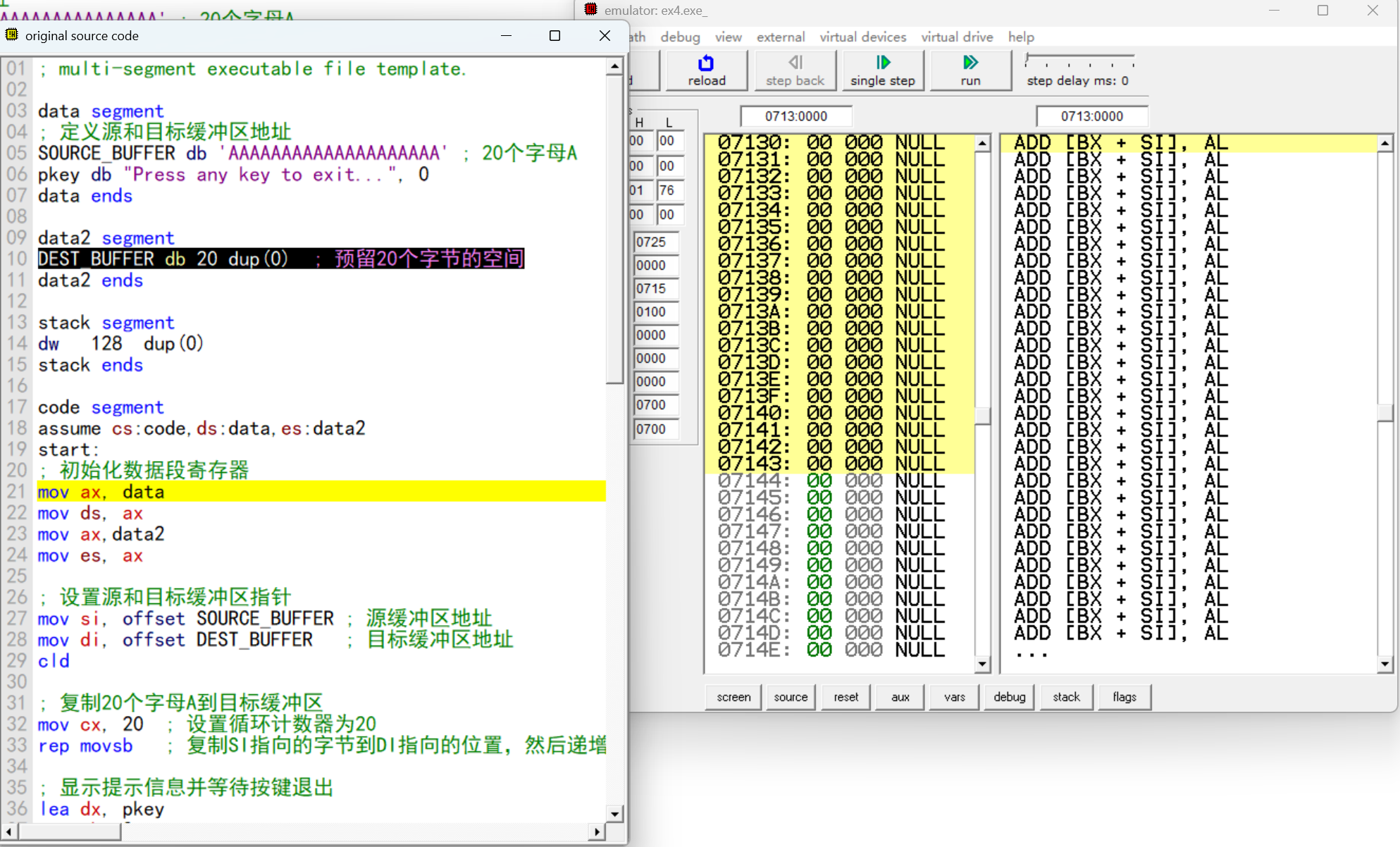


在debug下可以看到AX的值已经为X和Y之和

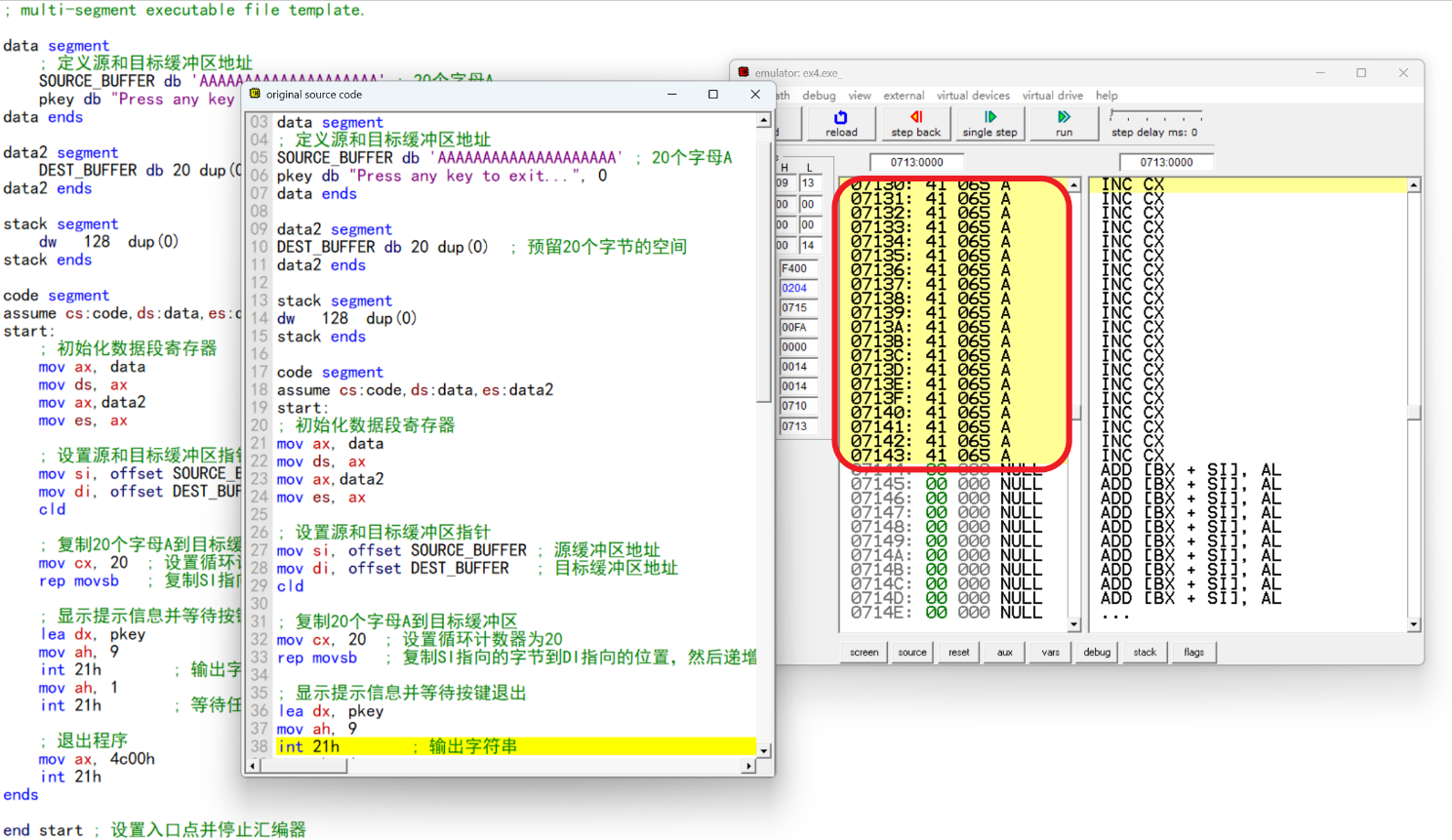
任务3：



代码



未复制前的目标地址

****

复制后的目标地址，可见复制成功

**三、设计思想**

绘制程序框图，并说明原理及算法、程序及数据结构、主要符号名等。

**原理及算法**

**原理**：  
程序的主要原理是内存操作和数据复制。在汇编语言中，程序通过操作内存地址和寄存器来执行指令，实现数据的复制和显示。rep movsb指令用于重复地复制从源地址（由si寄存器指定）到目标地址（由di寄存器指定）的字节，直到达到指定的次数（由cx寄存器指定）。

**算法**：

1. 初始化数据段寄存器（ds和es），以便访问数据段。
2. 设置源缓冲区（SOURCE\_BUFFER）和目标缓冲区（DEST\_BUFFER）的地址到si和di寄存器。
3. 将要复制的字节数（20）加载到计数器寄存器cx。
4. 使用rep movsb指令复制SOURCE\_BUFFER的内容到DEST\_BUFFER。
5. 显示提示信息并等待用户按键退出。

**程序及数据结构**

**程序**：  
程序由数据段、栈段和代码段组成。数据段包含SOURCE\_BUFFER、DEST\_BUFFER和pkey等变量。栈段用于存储程序的局部变量和返回地址。代码段包含程序的执行指令。

**数据结构**：

* SOURCE\_BUFFER：一个包含20个字母A的字节数组。
* DEST\_BUFFER：一个大小为20字节的未初始化数组，用于存储从SOURCE\_BUFFER复制过来的数据。
* pkey：一个字符串，用于在程序结束时显示提示信息。

**主要符号名**

**符号名**：

* SOURCE\_BUFFER：源缓冲区的标签名，存储要复制的数据。
* DEST\_BUFFER：目标缓冲区的标签名，用于存储复制后的数据。
* pkey：用于显示提示信息的字符串标签名。
* si：源索引寄存器，指向SOURCE\_BUFFER的起始地址。
* di：目标索引寄存器，指向DEST\_BUFFER的起始地址。
* cx：计数器寄存器，用于指定要复制的字节数。
* ds：数据段寄存器，存储数据段的基地址。
* es：附加段寄存器，通常与ds相同，用于某些字符串操作。
* ah：功能号寄存器，用于DOS中断调用。
* al、dl 等：其他通用寄存器，用于存储临时值或操作数。
* int 21h：DOS中断调用，用于执行各种系统服务，如输出字符串和退出程序。

**注释**

* 在编写程序时，需要确保所有使用的标签和变量都已经正确定义，并且寄存器和内存地址被正确初始化。
* rep movsb指令会自动递增si和di寄存器，以及递减cx寄存器，直到cx为0时停止复制。
* 程序通过DOS中断调用（如int 21h）来执行系统级别的操作，如输出字符串和退出程序。

综上所述，这个程序展示了如何在汇编语言中实现内存操作和数据复制，并通过DOS中断调用实现与用户的交互。通过理解程序的原理、算法、程序结构、数据结构和主要符号名，可以更好地理解和编写类似的汇编语言程序。

**四、程序代码**

程序清单。

Ex1：

; 程序功能是显示字符串 "hello world."

data SEGMENT ; 定义数据段

; 定义变量 message和pkey

; DB是字节类型

message DB "hello world.",0dh,0ah,"$"

pkey DB "press any key...$"

data ENDS

stack SEGMENT stack ; 定义堆栈段

DW 128 dup(0)

stack ENDS

code SEGMENT ; 定义代码段

assume CS:code,DS:data,SS:stack ;关联段寄存器和段名

start:

; data段名表示该段的段地址，赋值给数据段寄存器DS

MOV AX, data

MOV DS, AX

; 利用21H中断的9号功能，输出字符串message

LEA DX, message

MOV AH, 9

INT 21h

; 利用21H中断的9号功能，输出字符串pkey

LEA DX, pkey

MOV AH, 9

INT 21h

; 利用21H中断的1号功能，从键盘读取一个字符

MOV AH, 1

INT 21h

; 利用21H中断的4CH号功能，结束程序运行并返回操作系统

MOV AL, 0

MOV AH, 4ch

INT 21h

code ENDS

END start ; END伪指令指明此为源程序末尾，并指定程序入口是start标号

Ex2：

; multi-segment executable file template.

data segment

; 定义数据

X DW 1122H ; X的值

Y DW 3344H ; Y的值

Z DW ? ; 用于存储结果的Z单元

data ends

stack segment

dw 128 dup(0)

stack ends

code segment

assume cs:code,ds:data, ss:stack

start:

; 初始化数据段寄存器

mov ax, data

mov ds, ax

; 加载X和Y到寄存器

mov ax, [X] ; 将X的值加载到AX寄存器

add ax, [Y] ; 将Y的值加到AX寄存器中

; 将结果存储到Z单元

mov [Z], ax ; 将AX寄存器中的结果存储到Z单元

; 退出程序

mov ax, 4c00h

int 21h

code ends

END start ; 设置入口点并停止汇编器。

Ex3：

ORG 100h ; 设置程序起始地址

; 定义源和目标缓冲区地址

SOURCE\_BUFFER equ 200h

DEST\_BUFFER equ 300h

; 初始化源缓冲区为20个字母A

section .data

dbSource BUFFER 20 DUP('A')

section .text

start:

; 将源缓冲区地址加载到SI寄存器中

mov si, OFFSET dbSource

; 将目标缓冲区地址加载到DI寄存器中

mov di, DEST\_BUFFER

; 复制20个字母A到目标缓冲区

mov cx, 20 ; 设置循环次数为20

rep movsb ; 使用REP前缀和MOVSB指令复制字节

; 退出程序

mov ax, 4C00h

int 21h

end start

**五、结果分析**

1.程序的输出结果及对结果的分析。

程序的输出结果应为程序执行完毕后，在屏幕上显示的提示信息，例如“Press any key to exit...”。然而，对于汇编语言程序，我们不能直接观察其输出内容，而是通过观察程序是否成功地将SOURCE\_BUFFER中的数据复制到DEST\_BUFFER中来判断程序的执行结果。

通过调试器检查DEST\_BUFFER的内存地址，我们可以确认是否确实包含了20个字母A。如果DEST\_BUFFER中的内容与SOURCE\_BUFFER一致，那么说明程序成功执行了复制操作。此外，程序还应能够正常响应退出操作，即按下任意键后程序能够结束执行。

2.调试情况，如上机时遇到的问题及解决办法、观察到的现象及其分析。

在上机调试时，可能遇到的问题包括：

* 寄存器初始化不正确，导致复制操作失败。
* 内存地址访问错误，可能由于标签名或偏移量设置错误。
* 复制操作完成后，未正确设置退出条件，导致程序无法正常退出。

针对上述问题，解决办法如下：

* 仔细检查寄存器的初始值，确保si、di和cx被正确设置为源缓冲区地址、目标缓冲区地址和复制长度。
* 使用调试器查看内存地址的内容，验证是否按照预期进行复制操作。
* 在程序末尾添加正确的退出指令，如int 21h调用DOS中断以退出程序。

观察到的现象可能包括：

* 调试器中的内存视图显示DEST\_BUFFER的内容与SOURCE\_BUFFER一致。
* 程序在复制操作完成后，能够正确显示提示信息并等待用户按键退出。

对现象的分析：

* 如果DEST\_BUFFER的内容与SOURCE\_BUFFER一致，说明复制操作成功。
* 如果程序能够正常显示提示信息并退出，说明程序流程控制正确。

3.对程序设计技巧的总结及分析；实验的心得体会等。

在编写汇编语言程序时，需要注意以下几点技巧：

* 准确理解指令的功能和寄存器的作用，确保指令和寄存器的使用正确。
* 合理设置内存地址和标签名，避免地址冲突或访问错误。
* 注意程序的流程控制，确保按照预期的顺序执行指令。
* 利用调试工具进行调试，观察程序执行过程中的内存变化和寄存器状态，以便及时发现和解决问题。

实验的心得体会：

通过本次实验，我深入了解了汇编语言的基本原理和编程技巧。在实际编写和调试程序的过程中，我体会到了汇编语言编程的复杂性和挑战性。在遇到问题时，我学会了使用调试工具进行分析和定位问题，并逐步解决。通过这个过程，我更加熟悉了汇编语言的语法和指令集，提高了我的编程能力。

此外，我也认识到汇编语言编程需要耐心和细心，对指令和寄存器的使用需要准确无误。通过不断的实践和学习，我相信我会逐渐掌握汇编语言编程的精髓，并在未来的学习和工作中更好地应用它。