Hello World

同济大学 2022级 计算机科学与技术学院 软件工程专业 嵌入式系统方向 汇编语言课程作业

授课教师：王冬青

授课学期：2024-2025年度 秋季学期

2251730 刘淑仪

传统编译方式

环境配置

1. 安装 DOSBOX，安装完毕后，另外创建文件夹用于存放汇编语言所需要的编译配件。本人将其命名为 ASM_Tools 。
2. 在 DOSBOX 安装目录中，打开 DOSBox 0.74 Options.bat ，末尾添加如下内容设置编译虚拟环境。（此步骤非必要，但如果不做的话，需要每次启动 DOSBOX 都输入一遍该 mount 指令）

```
[autoexec]
# Lines in this section will be run at startup.
# You can put your MOUNT lines here.
mount d d:\ASM_Tools
d:
```

3. 在该文件夹下添加 LINK.EXE ， debug.exe ， MASM.EXE （均在学院服务器获得）

创建 .asm 文件

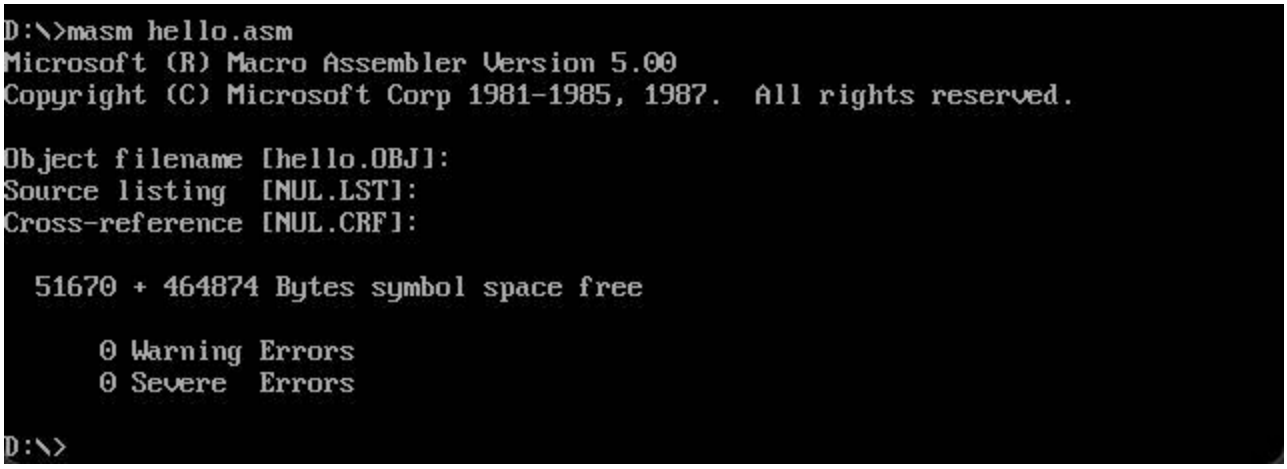
1. 在 ASM_Tools 文件下创建 hello.asm ，并添加简单的 hello world 程序（学院服务器也提供源代码）

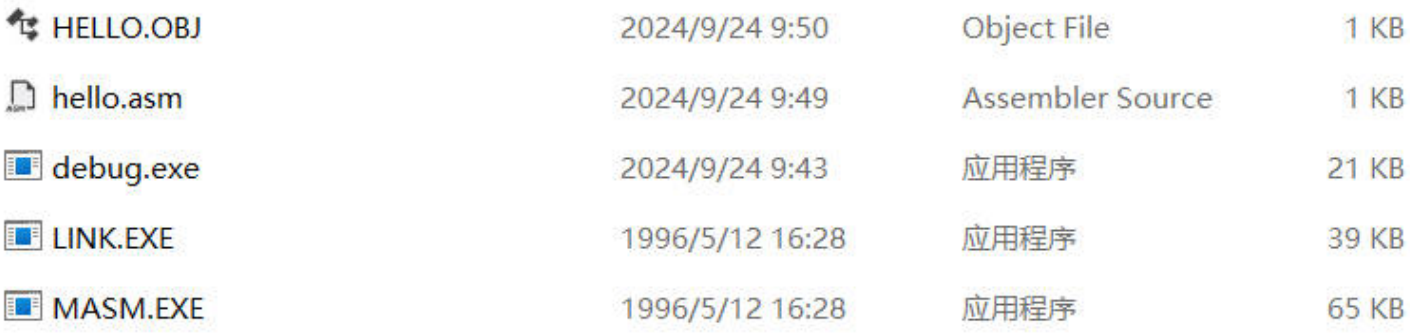
```
.model small
.data
Hello   DB 'Hello world!',0dh,0ah,'$'
.code
START:
        MOV     AX,@DATA
        MOV DS,AX
        MOV DX,offset Hello
        MOV AH,9
        INT     21H

        MOV     AX,4C00H
        INT 21h
END START
```

编译及运行 hello world 程序

1. 在 DOSBOX 中输入命令 `masm hello.asm` ，调用 MASM.EXE 编译程序，一路回车即可，得到 `hello.obj` 。





2. 在 DOSBOX 中输入命令 `link hello.obj` 或者直接 `link hello` ，调用 LINK.EXE 链接程序，同样一路回车即可，得到 HELLO.EXE 。

```
D:\>link hello.obj

Microsoft (R) Overlay Linker  Version 3.60
Copyright (C) Microsoft Corp 1983-1987.  All rights reserved.

Run File [HELLO.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment

D:\>
```

HELLO.EXE	2024/9/24 9:50	应用程序	1 KB
HELLO.OBJ	2024/9/24 9:50	Object File	1 KB
hello.asm	2024/9/24 9:49	Assembler Source	1 KB
debug.exe	2024/9/24 9:43	应用程序	21 KB
LINK.EXE	1996/5/12 16:28	应用程序	39 KB
MASM.EXE	1996/5/12 16:28	应用程序	65 KB

3. 输入命令 `hello`，执行 `HELLO.EXE`。

```
D:\>hello
Hello world!
```

4. 通过使用 `debug` 指令对 `EXE` 文件进行反汇编。在 `DOSBOX` 中输入 `debug hello.exe`，然后使用 `-u` 指令查看反汇编结果。

```
D:\>debug hello.exe
-u
076A:0000 B86B07        MOV     AX,076B
076A:0003 8ED8          MOV     DS,AX
076A:0005 BA0200        MOV     DX,0002
076A:0008 B409          MOV     AH,09
076A:000A CD21          INT     21
076A:000C B8004C        MOV     AX,4C00
076A:000F CD21          INT     21
076A:0011 004865        ADD     [BX+SI+65],CL
076A:0014 6C            DB      6C
076A:0015 6C            DB      6C
076A:0016 6F            DB      6F
076A:0017 20776F        AND     [BX+6F],DH
076A:001A 726C          JB      0088
076A:001C 64            DB      64
076A:001D 210D          AND     [DI],CX
076A:001F 0A24          OR      AH,[SI]
-_
```

内存写入数据方式

学习学院服务器下的 **hello剖析.pdf** 文件中对hello的另类执行方式，并在本地复现。

使用 debug 并查看寄存器

输入 `debug hello.exe`， `-r` 命令查看寄存器。

```
D:\>debug hello.exe
-r
AX=FFFF  BX=0000  CX=0021  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 B86B07        MOV     AX,076B
- ;
```

写数据及机器码到内存

1. 写数据：使用 `-e 076a: 0` 命令，将“Hello$"对应的ASCII码 `48 65 6c 6c 6f 24` 写入内存。

```
-e 076a: 0
076A:0000  B8.48   6B.65   07.6c   8E.6c   D8.6f   BA.24_
```

2. 写机器码：使用 `-e 076b: 0` 命令，将代码的机器码

   `b8 6a 07 8e d8 b4 09 ba 00 00 cd 21 b8 00 4c cd 21` （17个字节）写入内存。

```
-e 076b:0
076B:0000  21.b8   00.6b   48.07   65.be   6C.d8   6C.ba   6F.02   20.00
076B:0008  77.b4   6F.09   72.cd   6C.21   64.b8   21.00   0D.4c   0A.cd
076B:0010  24.21
```

修改寄存器及执行

1. 通过 `-r` 查看及修改对应的寄存器

   其中 `DS` 为数据段， `CS` 为代码段，其余暂不重要。

```
-r
AX=FFFF  BX=0000  CX=0021  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 48            DEC     AX
-r cs
CS 076A
:076B
-r ds
DS 075A
:076A
-r
AX=FFFF  BX=0000  CX=0021  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0000   NV UP EI PL NZ NA PO NC
076B:0000 B86A07        MOV     AX,076A
-_;
```

2. 输入 `-g` 执行。

```
-g
Hello
Program terminated normally
```