汇编语言 作业5

1. 作业完成情况

git链接：<https://github.com/1092888175/Assembly-Homework-5>

C语言文件采用X86-64 C11标准格式编写，使用GNU C Compiler汇编和链接

汇编采用X86-64 Linux格式汇编编写，使用The GNU Assembler汇编，使用GNU C Compiler调用GNU Linker链接

文件夹内包含shell脚本用于汇编和链接

作业 宏汇编的使用和多模块实现

内嵌汇编C文件：test.c

内嵌汇编可执行文件：test

多模块C文件：main.c

多模块汇编源文件：stdio.asm

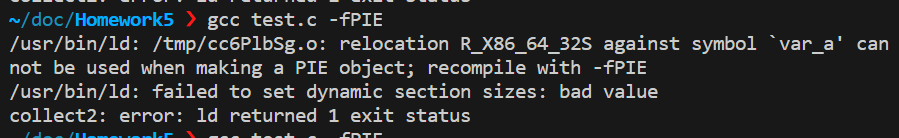
多模块目标文件：main.o stdio.o

多模块可执行文件：main

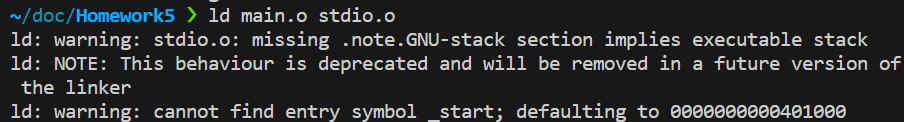
编译脚本：compile.sh

1. 开发心得和遇到的问题

在内联汇编开发中，使用基本汇编时会发生链接报错

猜测在64位C程序中，不能直接在内联汇编中访问变量，改用拓展内联汇编解决了问题

在多模块开发中，我首先将stdio.asm编译成stdio.o中间文件，再把main.c文件使用gcc生成目标文件main.o，然后使用ld命令将两个目标文件链接成可执行文件，在此过程中，遇到了如下问题



ld首先报了缺乏伪指令和弃用的警告，之后报了没有入口函数的警告。

在之前的开发中，ld默认使用一个全局的\_start函数作为程序入口，但可见C语言的main函数经过编译后不会产生\_start函数入口，通过使用gcc将main.c翻译成汇编文件，我们可以发现，其函数入口为全局的main，猜测由gcc调用ld会将默认入口改为main，因此我们使用gcc链接main.o stdio.o，成功消除了最后一条警告

对于前两条警告进行搜索，发现出于安全目的，Linux为了避免堆栈可被执行会发出这两条警告，我们需要手动使用noexecstack来加入这个安全功能

最终使用的命令序列

gcc main.c -c -g

as stdio.asm -o stdio.o -g

gcc main.o stdio.o -z noexecstack -o main