**汇编语言与逆向技术实验报告**

**Lab8 ARM平台-HelloWorld**

**学号：2112060 姓名：孙蕗 专业：信息安全**

1. **实验目的**
   1. 理解GNU ARM 汇编代码运行环境的搭建、配置及编译运行，掌握在华为鲲鹏云服务器上进行环境配置
   2. 命令行输出“HelloWorld”
2. **实验环境**

华为鲲鹏云主机、openEuler20.03 操作系统；

1. **实验内容**

以下步骤以在华为鲲鹏云服务器上执行为例。

1. **创建 hello 目录**

执行以下命令，创建 hello 目录，存放该程序的所有文件, 并进入 hello 目录。

mkdir hello

cd hello

1. **创建示例程序代码hello.s**

执行以下命令，创建示例程序源码 hello.s。

vim hello.s

代码内容如下：

.text

.global \_start

\_start:

mov x0,#0

ldr x1,=msg

mov x2,len

mov x8,64

svc #0

mov x0,123

mov x8,93

svc #0

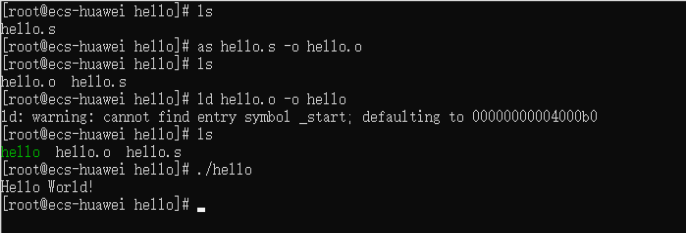
.data

msg:

.ascii "Hello World!\n"

len=.-msg

1. **进行编译运行**

保存示例源码文件，然后退出 vim 编辑器。在当前目录中依次执行以下命令，进行代码编译运行。

as hello.s -o hello.o

ld hello.o -o hello

./hello

通过上述代码运行，可以看出，编写的 hello-world 示例程序已经在华为鲲鹏云服务器上通过编译和运行，并成功输出结果。

1. **实验报告**
2. 汇编语句的解析

.text是处理器开始执行代码的地方，指定了后续编译出来的内容放在代码段可执行

.global \_start让\_start成为可见的标识符，给代码开始地址一个全局标记\_start，这样链接器就知道跳转到程序中的什么地方并开始执行程序

\_start是一个函数的起始地址，也是编译链接后程序的起始地址

mov x0,#x0 清零赋零存到x0

ldr x1,=msg把msg地址传到寄存器中

mov x2,len 字符串字符长度

mov x8,64 x8间接结果寄存器，写是系统调用#64

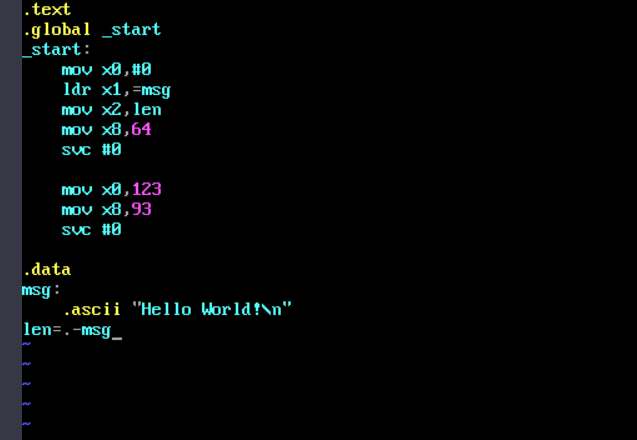
svc #0 系统调用#0

mov x0,123 退出

mov x8,93 退出位于内核函数表索引93处

svc #0 系统调用退出

定义字符串，并获取字符串长度



编译为二进制文件

链接为可执行文件

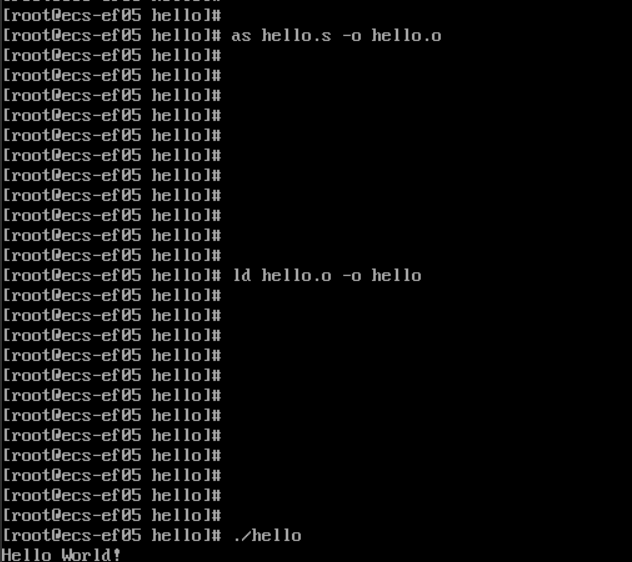
执行

as hello.s -o hello.o

ld hello.o -o hello

./hello

1. 运行截图



1. 回答问题：同样的代码能否在x86平台运行，为什么？

答：不能。

x86与arm使用的指令集，寄存器不同。