

# PA 2-1 实验报告

---

221220085 时昌军

## 一、实验目的

---

在完成了基本的运算功能后，我们希望计算机能够按照我们的命令来执行各种运算。在这一阶段，就要赋予NEMU这样的能力。



## 二、实验心得

---

本次实验真的好难。PA课上，宏的定义和各种指令操作就听的云里雾里，写的时候更是费了好一番功夫才堪堪理解。其中Segmentation Fault更是让人抓狂不已。还好有黄金版本可以用来debug。比较难的是call,ret,lea等指令。在经历pa2-1之后，我对课本上的指令架构有了更深的理解，也算是收获满满。总之，是很让人难忘的一次实验。

## 三、思考题

---

1. 使用 `hexdump` 命令查看测试用例的文件，所显示的文件的内容对应模拟内存的哪一个部分？指令在机器中表示的形式是什么？

答：查看add文件内容如下：

```

pa221220085@d16a7817ae92:~/pa_nju$ cd ./testcase/bin
pa221220085@d16a7817ae92:~/pa_nju/testcase/bin$ hexdump add.img
00000000 00e9 0000 5500 e589 ec83 c710 f445 0000
00000010 0000 45c7 00fc 0000 eb00 c748 f845 0000
00000020 0000 34eb 458b 8bfc 8514 3000 0003 458b
00000030 8bf8 8504 3000 0003 0c8d 8b02 f445 508d
00000040 8901 f455 048b 2085 0330 3900 74c1 b806
00000050 0001 0000 ff82 f845 458b 83f8 07f8 c476
00000060 45ff 8bfc fc45 f883 7607 83b0 fc7d 7408
00000070 b806 0001 0000 8382 f87d 7408 b806 0001
00000080 0000 b882 0000 0000 b882 0000 0000 c3c9
00000090 0000 0000 0000 0000 0000 0000 0000 0000
*
00010000 0014 0000 0000 0000 7a01 0052 7c01 0108
00010010 0c1b 0404 0188 0000 001c 0000 001c 0000
00010020 efe5 ffff 008b 0000 4100 080e 0285 0d42
00010030 0205 c587 040c 0004 0000 0000 0000 0000
00010040 0000 0000 0000 0000 0000 0000 0000 0000
*
00030000 0000 0000 0001 0000 0002 0000 ffff 7fff
00030010 0000 8000 0001 8000 fffe ffff ffff ffff
00030020 0000 0000 0001 0000 0002 0000 ffff 7fff
00030030 0000 8000 0001 8000 fffe ffff ffff ffff
00030040 0001 0000 0002 0000 0003 0000 0000 8000
00030050 0001 8000 0002 8000 ffff ffff 0000 0000
00030060 0002 0000 0003 0000 0004 0000 0001 8000
00030070 0002 8000 0003 8000 0000 0000 0001 0000
00030080 ffff 7fff 0000 8000 0001 8000 fffe ffff
00030090 ffff ffff 0000 0000 fffd 7fff fffe 7fff
00030a0 0000 8000 0001 8000 0002 8000 ffff ffff
00030b0 0000 0000 0001 0000 fffe 7fff ffff 7fff
00030c0 0001 8000 0002 8000 0003 8000 0000 0000
00030d0 0001 0000 0002 0000 ffff 7fff 0000 8000
00030e0 fffe ffff ffff ffff 0000 0000 fffd 7fff
00030f0 fffe 7fff ffff 7fff fffc ffff fffd ffff
0003100 ffff ffff 0000 0000 0001 0000 fffe 7fff
0003110 ffff 7fff 0000 8000 fffd ffff fffe ffff
0003120

```

根据NEMU，这部分内容将放到虚拟内存0x30000处（此时pa2-2还没做，ELF未装载），之后NEMU初始化会做两件事情：1.把测试用例镜像文件的内容直接拷贝到内存从0x30000处开始的连续区域内；2.将EIP初始化为0x30000。

指令在机器中以二进制编码的形式表示。

## 2. 如果去掉 `instr_execute_2op()` 函数前面的 `static` 关键字会发生什么情况？为什么？

会发生编译报错的情况。因为`instr_execute_2op()`函数被多重定义。

`static` 声明的函数为静态函数。`static` 关键字修饰后，函数的作用域就仅在本文件内，其他的文件都无法访问，该静态函数只能被本文件中函数调用，而不能被同一程序中的其他文件中的函数调用。在 NEMU不同指令的 `.c` 文件中都要实现 `instr_execute_2op()`，如果不声明为 `static`，其他文件中的函数也能调用 `instr_execute_2op()` 函数，从而导致 `instr_execute_2op()` 函数被多次定义。

## 3. 为什么 `test-float` 会 `fail`？以后在写和浮点数相关的程序的时候要注意什么？

使用浮点栈储存浮点数时为10字节，而`float`类型所占空间为4字节，因此在内存单元和浮点栈之间进行数据传送的过程中可能会发生精度损失，从而造成计算结果错误。

以后在写和浮点数相关的程序的时候要注意浮点数能否精确表示，要注意各个部分之间的精度转换。