PA3实验报告

姓名：何伟　　**学号：**171240537

2018 年 11 月 22 日

摘要

完成pa3中的必做内容。

1　实验进度

已完成所有必做内容，最后运行了仙剑奇侠传。

2　思考题

PAL_LoadGame()中调用fread函数读取游戏数据，fread在libs文件夹的libc中，它会调用_fread_r函数，最终通过调用memcpy函数处理文件中的数据，文件通过已经fopen打开。
redraw()函数通过调用NDL_DrawRect()更新屏幕，最终会调用fwrite库函数进行屏幕信息的写入，fwrite会经过一系列的函数调用最终调用sys_call，通过传入的参数判断为sys_write，而sys_write判断文件的信息，文件的写指针不为空，通过函数指针调用fd_write,再调用draw_rect调用进行写的操作。
总体看来，再nanos-lite中执行会进行由边函数的编译并将navy-apps中的文件放入指定位置，make run执行后根据proc.c中定义的文件通过loader.c载入，navy-apps中进行了系统调用有关的前期处理，库函数的实现在navy-apps/libs中，通过libos/nanos.c中的函数进行系统调用，之后由操作系统进行文件的处理，之后会调用nexus-am中的函数进行硬件方面的操作。而这些的执行都是在x86模拟器nemu上实现的。

3　实验心得及遇到的问题

3.1　PA3.1 穿越时空的旅行

开始总是困难的，从PA2的指令实现到PA3的操作系统。阅读源码和讲义占了PA3.1的大部分时间，之后的实现实现起来倒也没有遇到特别大的问题。不过在重新组织_Context时没有进行验证就继续往下做了，“感觉”应该没问题，做到后面果然还是出了问题，还是要一步一步慢慢来才行。

3.2 PA3.2 用户程序和系统调用

在第一步便遇到了难题，实现loader.c时nemu正常运行但是native不行，后来还是在于同学交流的时候意识到了问题，一开始理解错了文件读写的含义，并且我是通过文件开始位置和0x4000000偏移量确定位置进行读的操作，但是native和nemu的偏移量并不相同，后来直接读到0x4000000native和nemu就可以正常运行了。
之后是在实现系统调用时遇到了问题，首先时不明白个中“返回值”。刚开始写的时候并没有特别理解源代码，有点混乱，不知道讲义中说的是nanos.c中的返回值，还是sys_call中的返回值，当时也就凭着感觉瞎写了，反正不会就改一个返回值。有一天在大佬的指导下，发现nanos.c中的sys_call函数是有返回值，就是设在sys_call.c中的返回值，原来是一个东西，这是后话。
中间还遇到一个想起来非常心痛的事(可能花了五六个小时)。在实现sys_write时始终只能输出第一句话，后面的printf无法执行，当时加了一堆调试信息，胡乱输出。于是胡乱调试，试过把nanos.c中的exit删掉，但是并没有发现输出了printf中的内容，反而时调试的信息变得。知道最后才发现，每句调试内容前面的第一个字母正好是printf中的内容!!! 还真是要把exit删掉，看到后面的讲义，才知道没有实现缓冲区之前就是一个一个字符输出的，好吧，跪了Orz。还是怪自己并没有深入理解系统调用的过程，不清楚exit什么时候该用。

3.3 PA3.3 文件系统

PA3.3快做哭了233333。先是实现一写简单的文件操作，好像写得挺顺利，text测试也过了，没有花多少时间。实现VFS，什么意思啊，函数指针好像也没有理解用途，不知道怎么用(后来靠大佬指点明白了)，所有的函数调用都在fs_xxx里面通过判断直接写了，没用函数指针。将VGA显存抽象成文件，花了好久找屏幕的宽度和高度怎么获取。反正可选信息也不过，排列组合了，终于试除了图案，可是怎么这么小啊，不知道对不对，反正就继续做了。后面的events也一会实现了。激动人心的时刻到了，开始跑仙剑，晚上十点，native试水，成功开始游戏，啊，激动啊。nemu运行，画面成功加载，哭了，好感动啊。开始游戏，退出??? hal.c里面触发了assert(0)。于是，便开始了漫长的debug之旅。一时间想不通啊，于是尝试着读pal的代码，读不通啊，只能读个大概，assert也加了不少，好像还是不知道哪里错了。于是到前面看吧，是不是要开diff跑一下，肯定不能跑仙剑，于是选择了前面的text，漫长的等待，diff跑出来了!!! 报错了，于是一步一步跟踪啊。终于找到了罪魁祸首，int后面接了一个push指令出错了，算了一个怎么试push的eflags啊，算是指令实现里面最模糊的地方了，搞了一个小时之后，发现了是少了AF位，不是吧，这个寄存器根本没用过啊，仔细一下，好像qemu和nemu在eflags这块处理不太一样吧，似乎不能用diff了，哭了 (据说凌晨的git记录有加分)。第二天继续debug，完全不知道哪里错了啊，无从下手。白天全在debug，验证感觉可能错的地方，自闭了。到了晚上，在大腿的指导与帮助下，终于de出来了，原来是add错了，哭了。终于能玩仙剑了，还是有点开心的。

3.4 心得体会

前面做得太快，读完讲义就开始做，不会再读一遍，并没有深入的思考，像前面提到的删除exit的问题，就是因为对系统调用的执行过程没有深入理解，走了不少弯路。在做的时候有事会跟大腿交流遇到的bug，好像有好几个人说到忘了考虑指令为16的情况，突然意识到，好像前面实现指令的时候有些指令也是只写了32位，感觉不会用到16位，想起来真是太危险了，这边发生bug可能又要花很长时间，于是在没有考虑16位的地方加了assert，后面遇到了也容易定位错误。另外，感觉debug的能力太低了，有时还可能被自己写的调试信息绕晕。一个小的错误一花就是几个小时的时间，能力有待提高。

写在最后：感谢李顶为大腿帮我debug。顺便贴上玩仙剑的照片23333

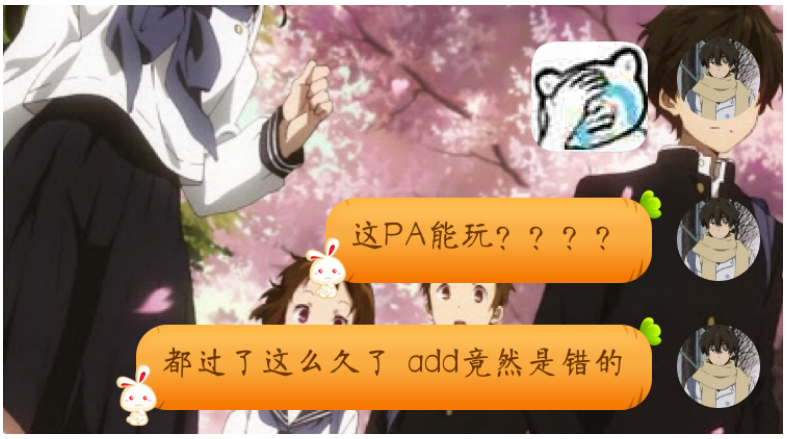

图 1: 得知add是错的

图 2: 运行仙剑

图 3: 运行仙剑

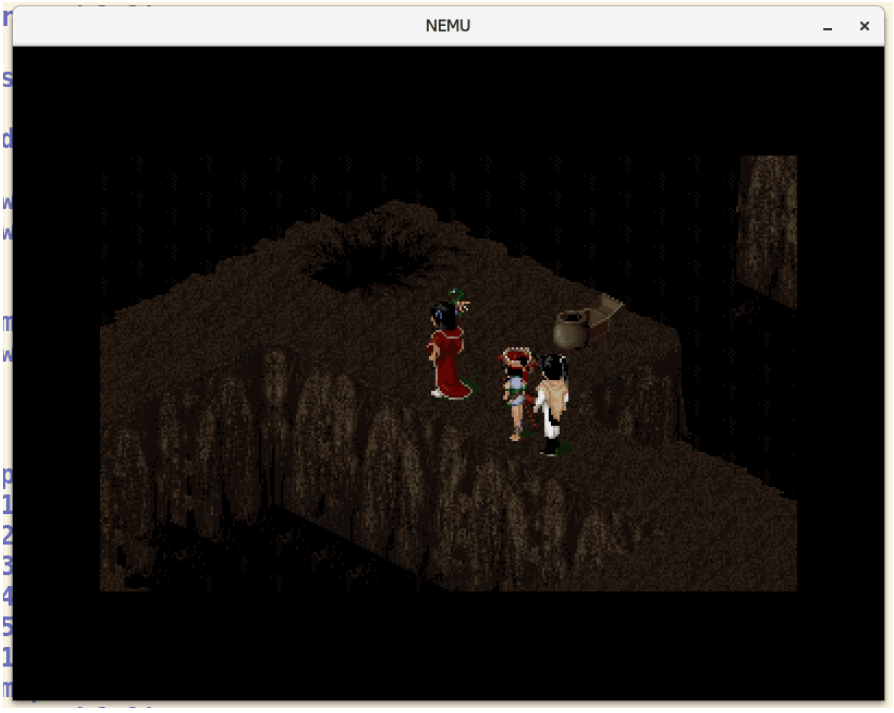

图 4: 运行仙剑