华中科技大学

2022

系统能力培养　课程实验报告

题　　目：	riscv32 模拟器设计
专　　业：	计算机科学与技术
班　　级：	CS2010 班
学　　号：	U202015583
姓　　名：	穆嘉毅
电　　话：	18840781565
邮　　件：	695693802@qq.com
完成日期：	2024-1-15

计算机科学与技术学院

目 录

1　课程实验概述

1.1 课设目的

在代码框架中实现一个简化的 RISC-V 模拟器

可解释执行 riscv32 执行代码

支持输入输出设备

支持异常流处理

支持精简操作系统

支持文件系统

支持虚存管理

支持进程分时调度

最终在模拟器上运行“仙剑奇侠传”,让学生探究“程序在计算机上运行”的机理，掌握计算机软硬协同的机制，进一步加深对计算机分层系统栈的理解，梳理大学 3 年所学的全部理论知识，提升学生计算机系统能力。

2　实验方案设计

2.1 综述

1）世界诞生前夜---开发环境： 安装虚拟机或者 docker，熟悉相关工具和平台，安装 sourceinsight 阅读代码 框架。

2）开天辟地---图灵机

PA1.1 简易调试器

PA1.2 表达式求值

PA1.3 监视点与断点

3）简单复杂计算机--冯诺依曼计算机

PA2.1 运行第一个 C 程序

PA2.2 丰富指令集，测试所有程序

PA2.3 实现 I/O 指令，测试打字游戏

4)穿越时空之旅：异常控制流

PA3.1 实现系统调用

PA3.2 实现文件系统

PA3.3 运行仙剑奇侠传 2

5）虚实交错的魔法：分时多任务

PA4.1 实现分页机制

PA4.2 实现进程上下文切换

PA4.3 时钟中断驱动的上下文切换

2.2 PA0

搭建开发环境时，我选择了在电脑上装了 linux 系统来允许本实验的代码，平台为 GNU/Linux + gcc + C。首先用一个空 U 盘作为系统启动盘下载 linux 系统，

之后划分 Windows 的一部分空间进行磁盘分区来作为 linux 的空间，重启电脑进入 linux 系统并进行初始化。之后再利用 apt-get 命令安装一些工具包，并 git clone 下来项目的仓库，进行初始化，完成 PA0 部分。

2.3 PA1

task PA1.1: 实现单步执行, 打印寄存器状态, 扫描内存
task PA1.2: 实现算术表达式求值
task PA1.3: 实现所有要求, 提交完整的实验报告

PA1.1

完成这一部分的任务要先通读一下框架代码，对项目的整体架构有一个足够的认识。而这一 PA 我们需要了解的项目代码只有 nemu 部分，NEMU 主要由 4 个模块构成: monitor, CPU, memory, 设备。而由于需要实现单步执行，打印寄存器状态，扫描内存这三个函数。在文件 ui.c 中这三个函数被定义为 cmd_si，cmd_info 和 cmd_x。cmd_si 调用 cpu_exec 执行指定的步数，cmd_info 调用另外一个函数进行寄存器名字与存储的值的输出， 这个函数的名字是 isa_reg_display，而它又使用函数 reg_name 和宏 reg_l 进行寄存器名称与值的打印， cmd_x 中对输入的参数进行分割，用 vaddr_read 函数读出对应地址的值然后打印出来。

PA1.2

本节的任务是完成表达式求值。首先将词法分析规则和对应的正则表达式添加到 expr.c 文件中，然后完成 make_token 函数，该函数用于将其他类型的标记添加到算术表达式中。 该规则在成功识别出 token 后，将 token 信息依次记录在 token 数组中。 然后用 check_paentheses 函数填充相应的括号，并完成 op_priority 运算符的优先级函数，实现根据 C 语言优先级实现相应的运算符优先级。 实现这些辅助函数后，可以创建递归评估函数。 评估输入参数 p 和 q 的过程。如果 p>q，则立即返回 0，并将 success 设置为 false。如果 p==q，则令牌类型为整数、十六进制或寄存器。 如果是，则返回该值。 否则，成功为 false，并返回 0。 然后它检查括号是否匹配，如果匹配， eval(p+ 1,q) 返回 -1 ，否则继续搜索基本运算符。 如果找到基运算符，则返回根据基运算符对应的值。 如果没有找到，4 成功返回 false0。 eval 函数完成后，将 eval 和 make_token 函数合并到 expr 函数中，完成整个表达式求值范围的写入。

PA1.3

本节的任务是完成实现监视点功能，首先完善结构体 watchpoint，结构体定义在 nemu/include/monitor/watchpoint.h 中，添加相应的成员变量， 而为了监测表达式，需要一个 char 数组存放表达式，同时为了监测值的变化，需要用一个变量用于存储旧值。之后在文件 watchpoint.c 中实现函数 new_wp，free_wp，print_wp。new_wp 用于新建一个监视点，把链表 free_中的节点放入链表 head 中；free_wp 用于释放监视点，根据编号在 head 链表中找到相应的节点， 放入链表 free_，这两个函数的操作都是通过链表的插入与删除来实现的，为了方便快捷在插入链表时都选择头插法。print_wp 函数则用于打印监视点信息。完成上述函数后，在 ui.c 中使用上述函数完成命令 cmd_x 与

cmd_d， 完善函数 cmd_info，同时要在 cpu.exec 中实现当监视点状态改变使程序暂停执行的逻辑，该逻辑较为简单，程序每执行一步就对所有的监视点进行表达式求值， 并与旧值进行比较，如果两值不同，则置 nemu 的状态为 NEMU_STOP。

2.4 PA2

task PA2.1: 在 NEMU 中运行第一个 C 程序 dummy
task PA2.2: 实现更多的指令, 在 NEMU 中运行所有 cputest 23:59:59
task PA2.3: 运行打字小游戏, 提交完整的实验报告

PA2.1

这一部分要求我们运行 dummy 程序，第一次直接使用 make 命令跑 dummy 程序，会显示 abort，随后会在 build 文件夹中生成对应的反汇编文件，查找手册获取未实现的指令，最后可以确认 dummy 要实现的新指令包括 auipc，addi，jal， jalr，然后在 all-instr.h 文件中定义对应的执行函数指令，在 exec.c 文件中的 opcode_table 添加新指令和对应的实现。文件 decode.c 中实现一些工具函数并修改完善 rtl.h 中的 RTL 指令，利用 Calculate.c、control.c 等文件中的 RTL 指令实现工具函数。 之后，再次编译并运行。

PA2.2

该过程与 PA2.1 类似，只是需要实现更多指令。 更多的指令意味着更多的错误，所以最好先做一下差异测试。 diff 测试在指令执行期间比较独立实现的 Nemu 和 Qemu，并在每个执行阶段比较两者中的 32 个寄存器的值是否相同，否则会报错。 使用 PC 拆解值，可以找出反汇编代码中的错误行，然后对其进行分析并找到解决方案。 差异测试完成后，就可以开始执行说明。 可以将测试文件从小到大排序，逐一编译运行，找到未实现的指令来执行。另外，测试文件 hello-str 和 string 使用了 sprintf、strcmp、strcat、strcpy 和 memset 库函数，必须写在 nexus-am/libs/klib/src/stdio.c 和 string.c 中。在 string.c 中需要编写与常用字符串相关的函数，比较简单，如 strcmp、strlen、strcat 等。在 stdio.c 中需要编写 sprintf、printf 等函数，这两个函数都经过中间过程功能。vsprintf 需要处理传入的字符串 fmt 和变量参数列表 va_list。 当从 FMT 字符串中读取子字符串%d、%x 和%s 时，调用 va_arg 函数将对应的参数类型更改为 va_list。 ，然后将其转换为字符串并将其传递给输出字符串以完成 vsprintf 功能。 只需在 sprintf 和 printf 函数内调用 vsprintf 函数即可执行相应的函数。在 PA2.2 的所有内容全部完成后，所有程序都可以进行一键回归测试来验证通过情况。

PA2.3

这一模块要求完成串口、时钟、键盘、VGA 四种输入输出设备程序的编程。串口在 trm.c 文件中实现。 在此之前，我们还需要编写一个 printf 函数来让程序运行。 文件 nemu-timer.c 中的时钟功能需要改进。 当开始初始化时，可以通过 RTC_ADDR 获取开始时间。 当运行时钟函数时，将在 RTC_ADDR 处获取当前时间。 将 uptime->hi 设置为 0，将 uptime->lo 设置为当前时间。时间与开始时间之差即可完成时钟功能。 文件 nemu-input.c 中的键盘功能需

要改进。 在 KBD_ADDR 处获取键盘按键信息，放入 kbd->keycode 中，并与 KEYDOWN_MASK 比较按键信息。 如果值为 1，则表示按下，值为 0，必须解析指令。 最后要实现的是 VGA 设备。 nemu-video.c 中应完善相应功能。VGA 设备必须将逐像素信息写入 VGA 对应的地址空间中，才能正确显示图像。

2.5 PA3

task PA3.1: 实现自陷操作_yield()及其过程
task PA3.2: 实现用户程序的加载和系统调用, 支撑 TRM 程序的运行
task PA3.3: 运行仙剑奇侠传并展示批处理系统, 提交完整的实验报告

PA3.1

这一部分的任务是要实现自陷操作，必须先实现自动捕获指令。 就需要实现 csrrs、csrrw、ecall 和 sret 指令。 自陷操作是使用 Ecall 实现。 csrrs 和 csrrw 指令修改控制状态寄存器。 sret 指令用于自陷后返回，之后要在 intr.c 文件中编写 raise_intr 函数。 raise_intr 函数用于模拟相应的过程：将当前 PC 值保存在 SEPC 寄存器中，在 Scuse 寄存器中设置异常编号，从 STVEC 寄存器中取出异常写入地址，并访问异常写入地址。 执行完拦截操作后，需要保存上下文，并按照拦截器汇编代码的顺序恢复_Context 成员的基本结构。然后 S 执行正确的事件。 传播需要通过__am_irq_handle 函数中的异常号来识别捕获异常，在 do_event 函数中识别捕获事件_EVENT_YIELD，最后使用 sret 语句返回并恢复上下文。

PA3.2

这一部分的任务是要实现用户程序的加载和系统调用, 为了支持 TRM 程序执行，首先必须实现加载函数来加载用户程序。 由于文件系统尚未实现，可执行文件的代码和数据直接存储在相应的内存中。通过 load 函数中的 ramdisk_read 函数找到它，然后进入程序入口。 接下来，需要完成系统调用的执行。 这类似于 PA3.1 中识别陷阱事件，因此 Nemu 可以识别系统调用。 然后在 do_event 中添加 do_syscall 调用。根据 nanos.c 文件中的 ARGS_ARRAY，在 riscv32-nemu 中实现正确的 GPR 宏，然后添加 SYS_yield、SYS_read、SYS_write 和 SYS_brk 系统调用，并完成_sbrk 函数来实现堆区域管理。

PA3.3

这一部分的任务是要实现文件系统，添加设备的支持，最终能够运行仙剑奇侠传。首先，需要实现 fs_open、fs_read 和 fs_close 函数，因为虚拟磁盘上的文件数量增加后，在 load 函数中直接使用 ramdisk_read 已经不再合适了。 完成这些 fs 函数后，替换 loader 函数中的 ramdisk_read 函数。 修改逻辑，以 Loader 中的文件名来表示加载的程序，然后运行 fs_write 和 fs_lseek 函数进行输出。 这些功能完成后，还应该添加相应的系统调用； 为了实现虚拟文件系统，VFS 将 IOE 抽象为文件。 首先，VFS 中需要添加对各种特殊文件的支持。 然后需要实现 serial_write 函数来完成串口的写入。 那么 events_read 的执行就完成了，支持了读操作。最后需要完成 init_fs、fb_write 和 fbsync_write 函数。 、init_device、dispinfo_read 来支持 VGA 设备。如果没有发生错误，就可以运行仙剑奇侠传了。

3　实验结果与结果分析

3.1 PA1

:

进入调试器，输入 help

```
[src/monitor/monitor.c,36,load_img] No image is given. Use the default build-in
image.
[src/memory/memory.c,16,register_pmem] Add 'pmem' at [0x80000000, 0x87ffffff]
[src/monitor/monitor.c,20,welcome] Debug: ON
[src/monitor/monitor.c,21,welcome] If debug mode is on, A log file will be gener
ated to record every instruction NEMU executes. This may lead to a large log fil
e. If it is not necessary, you can turn it off in include/common.h.
[src/monitor/monitor.c,28,welcome] Build time: 15:48:39, Jan 14 2021
Welcome to riscv32-NEMU!
For help, type "help"
(nemu) help
help - Display informations about all supported commands
c - Continue the execution of the program
q - Exit NEMU
si - Single Step Execute
info - Print details of register || watchpoint
x - Scan memory
p - Expression Evaluation
w - Set a New Watchpoint
d - Delete Watchpoint
(nemu)
```

W 命令是设置监视点

```
(nemu) w $t3
Set Watchpoint Succeed
```

Info 命令是查看细节

```
(nemu) info w
NO          EXPR                    VALUE
3           $t3                     0
```

Si 命令是单步执行

```
(nemu) si 2
80100004:   23 a0 02 00                         sw  0(t0),$0
80100008:   03 a5 02 00                         lw  0(t0),a0
```

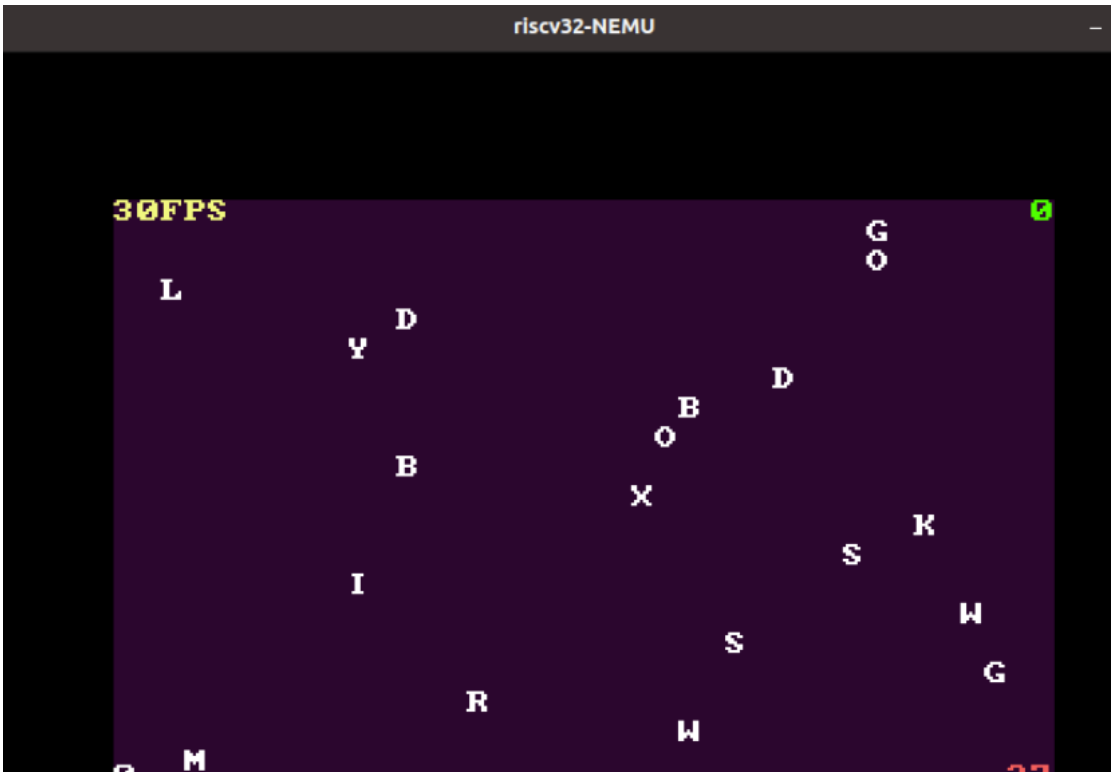
P 命令是对表达式进行计算，如果表达式错误会报错

3.2 PA2

可以通过一键回归测试：

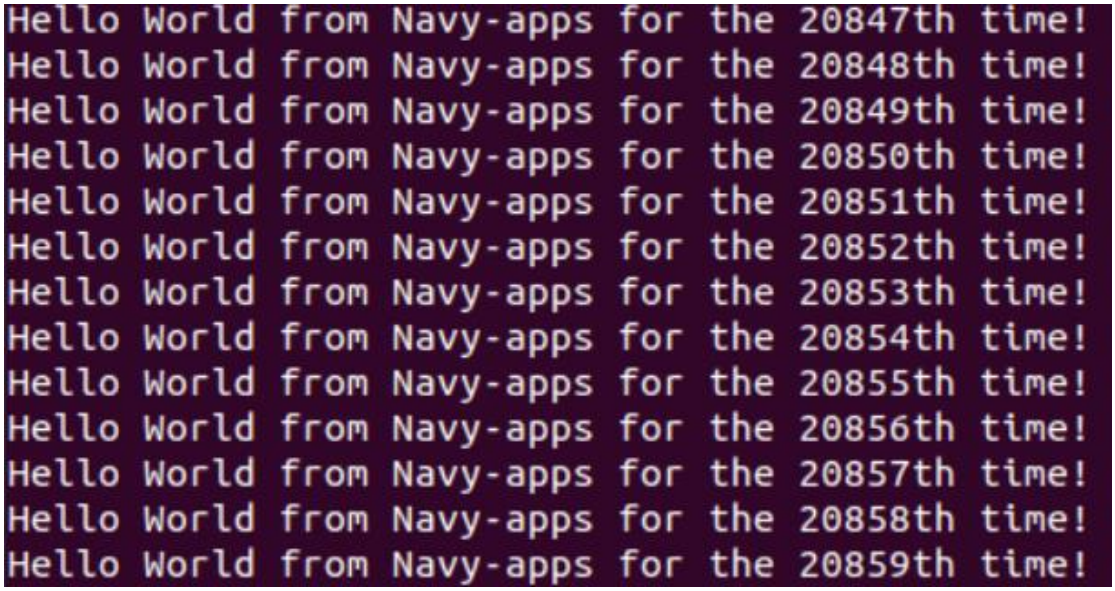
```
          div] PASS!
        dummy] PASS!
         fact] PASS!
          fib] PASS!
     goldbach] PASS!
    hello-str] PASS!
      if-else] PASS!
    leap-year] PASS!
   load-store] PASS!
   matrix-mul] PASS!
          max] PASS!
         min3] PASS!
        mov-c] PASS!
        movsx] PASS!
 mul-longlong] PASS!
       pascal] PASS!
        prime] PASS!
   quick-sort] PASS!
    recursion] PASS!
  select-sort] PASS!
        shift] PASS!
  shuixianhua] PASS!
       string] PASS!
 sub-longlong] PASS!
          sum] PASS!
       switch] PASS!
to-lower-case] PASS!
      unalign] PASS!
       wanshu] PASS!
```

可以正常运行打字小游戏：

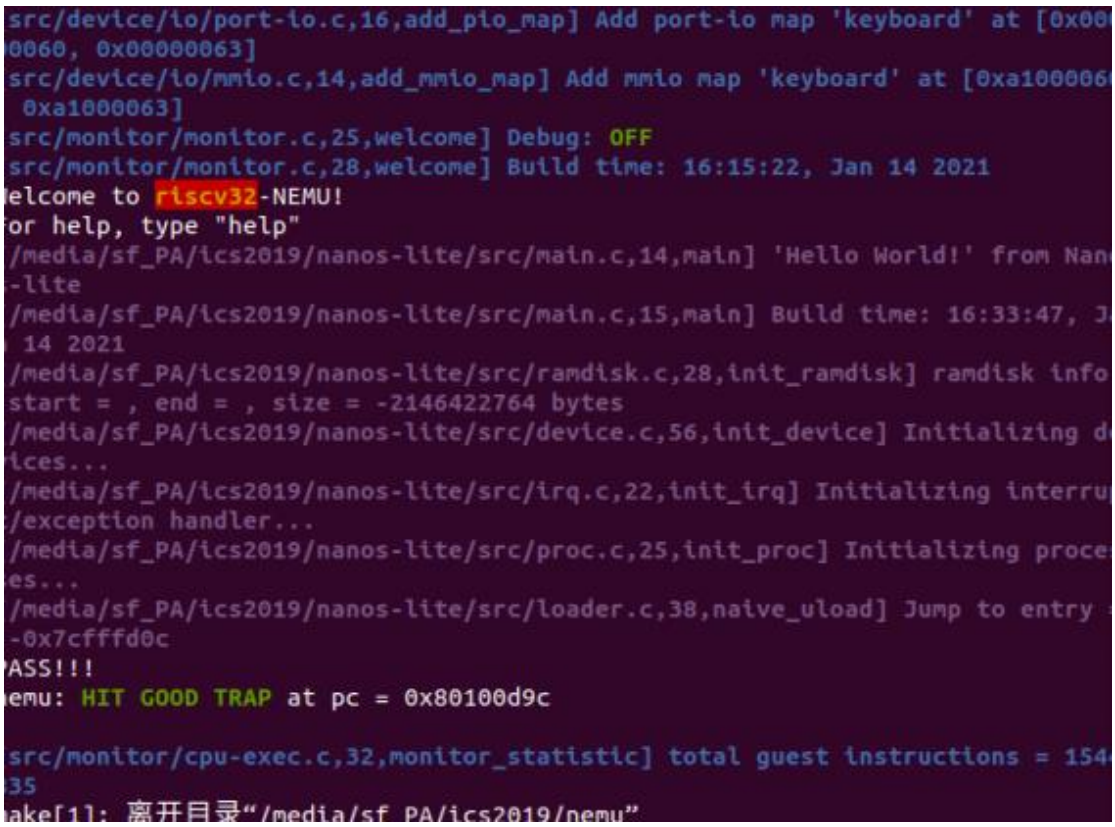


3.3 PA3

Hello 测试：



Text 测试：



Events 测试：

```
receive time event for the 1124352th time: t 26722
receive time event for the 1125376th time: t 26743
receive time event for the 1126400th time: t 26764
receive time event for the 1127424th time: t 26783
receive time event for the 1128448th time: t 26803
receive time event for the 1129472th time: t 26823
receive time event for the 1130496th time: t 26843
receive time event for the 1131520th time: t 26864
receive time event for the 1132544th time: t 26884
receive time event for the 1133568th time: t 26903
receive time event for the 1134592th time: t 26923
receive time event for the 1135616th time: t 26943
receive time event for the 1136640th time: t 26962
receive time event for the 1137664th time: t 26982
receive time event for the 1138688th time: t 27001
receive time event for the 1139712th time: t 27021
receive time event for the 1140736th time: t 27042
```

运行仙剑奇侠传：

4　实验总结与心得体会

本实验广泛地涉及到了计算机体系架构的各种知识，包括但不限于词法分析，语法分析，汇编反汇编，计算机组成原理的指令集… 总体来说，是一个对之前学习到的知识的一个整体的应用。同时，我也借此机会，第一次自己在 Windows 系统下安装了 linux 系统，之前要么是用虚拟机，要么是像在实习的时候直接用公司内网的工具远程连接服务器来操作。此外，这次实验也是涉及到了交叉编译相关的知识，还有引用第三方库的时候也有版本问题，至于 git 相关的项目管理之前也早就涉及到，就不再赘述。

然而，本次实验也存在一些客观上的问题。比如，本实验没有任何的理论教学，全靠自学，很容易遇到一些与实验内容无关的错误。比如，我一开始没细看群里的文档，靠着之前的模糊印象，直接 git clone 下来初始仓库开始做，后来发现 clone 错了版本，选择了 2021 的版本，幸好发现的时间尚早，而且内容大差不差，改回来的成本并不大。

同时，本实验设立的时间也有一定的问题。本实验是设置在了大四的上学期，这是本科阶段四年中不说最忙也是涉及到与上课外的事情最多的一个学期。有的同学要考研，而我也是先从实习岗位离职，然后又匆忙地参与秋招，在秋招差不多尘埃落定之后又要参与申请港校研究生的笔试面试的工作，导致投入到本实验上的时间没有太多而且断断续续，也没有心思把本实验尽量做到更好，因此只完成了基本的要求。

瑕不掩瑜，虽伴随着一些问题，本实验作为本科阶段毕业设计前的最后一门专业课，既是对过去学习的理论知识的一次实践，也是一个对我有着一定帮助的项目，同时因为有着一些阶段性的测试，也涉及到了一些对代码的调试工作，尤其又是一个代码量相对较大的项目，本实验也是十分规范而严谨的，总体上看，我在完成本实验后受益匪浅。

参考文献

[1] https://nju-projectn.github.io/ics-pa-gitbook/ics2019/

[2] 秦磊华，吴非，莫正坤.计算机组成原理. 北京：清华大学出版社，2011 年

[3] https://course1.istratus.cn/projects/pa/wiki