南京航空航天大学《计算机组成原理Ⅱ课程设计》报告

- 姓名：曹伟思
- 班级：1617302
- 学号：161730213
- 报告阶段：PA3.1
- 完成日期：2019.6.9
- 本次实验，我完成了所有内容。

目录

思考题

什么是操作系统?

操作系统,顾名思义就是用于操作的一个系统,其操作的对象是电子设备,比如读写硬盘上的数据,将程序加载进内存(创建进程)等等.操作系统抽象这些操作,以一种人们方便,简洁的操作接口展示出来,其更笨目的是为了帮助人们更简单操作电子设备.

我们不一样，吗?

没有什么差距,可以,两者之间知识功能不同.

操作系统的实质

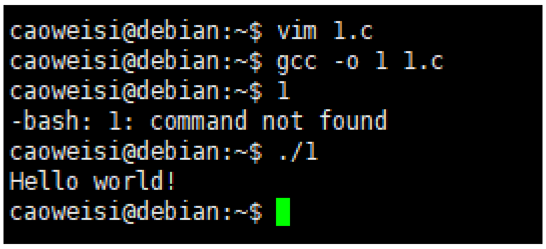
操作系统就是一个较为大型的程序,它和直接运行在硬件上的程序无实质差别.

程序真的结束了吗?

回答正确,实际上是返回到了`__libc_start_main`函数(`linux`).

触发系统调用

编译运行代码即可.



```
caoweisi@debian:~$ vim 1.c
caoweisi@debian:~$ gcc -o 1 1.c
caoweisi@debian:~$ 1
-bash: 1: command not found
caoweisi@debian:~$ ./1
Hello world!
caoweisi@debian:~$ 
```

有什么不同?

函数调用,调用者保护寄存器,可以,与用户编写的函数的不同点在于保存的内容和特权级不同.

段错误

段错误发生原因为程序访问了没有权限访问的虚拟内存页,而程序会访问哪些地方编译阶段是不可知的,段错误通常是由程序数据越界,使用了野指针,或者是使用了不安全的函数(`gets`)导致的.

对比异常与函数调用

因为这两种情况下会破坏的环境和恢复状态需要的环境不一样.

诡异的代码

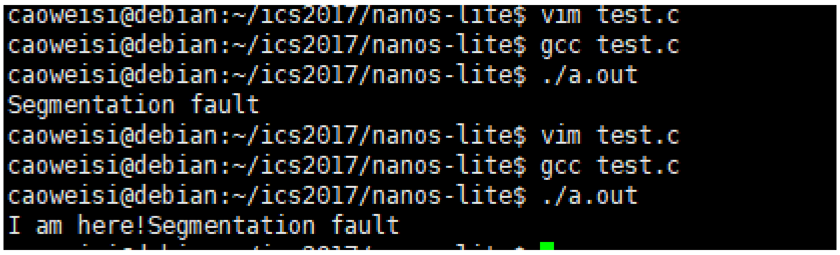
因为构造`trap frame`改变了`esp`.

注意区分事件号和系统调用号

事件号是异常中断的号码,系统调用号是当事件号为80触发系统调用时`eax`寄存器的值.

打印不出来?

`fflush(stdout)`刷新缓冲区即可.



```
caoweisi@debian:~/ics2017/nanos-lite$ vim test.c
caoweisi@debian:~/ics2017/nanos-lite$ gcc test.c
caoweisi@debian:~/ics2017/nanos-lite$ ./a.out
Segmentation fault
caoweisi@debian:~/ics2017/nanos-lite$ vim test.c
caoweisi@debian:~/ics2017/nanos-lite$ gcc test.c
caoweisi@debian:~/ics2017/nanos-lite$ ./a.out
I am here!Segmentation fault
```

理解文件管理函数

fs_open():用文件名参数到文件描述符表中匹配并返回下标,没找到则报错. fs_read():通过fd参数获取文件偏移和长度,再从ramdisk或dispinfo中读取数据到buf中. fs_write():通过fd选择写方式.若是文件写,则计算偏移和读取长度进行文件读写. fs_lseek():通过whence选择移动文件偏移的方式,然后将新的偏移赋给对应文件描述符. fs_close():直接返回0,因为不需要close.

不再神秘的秘技

因为写程序就是写bug.

必答题

编译后的程序被保存在ramdisk文件中,make run先运行nemu,然后在nemu上运行Nanos-lite.Nanos-lite的main函数中使用loader加载位于ramdisk存储区(实际存在与内存中)的/bin/pal程序.loader函数从ramdisk文件(磁盘)中读取程序到内存区,进行一些初始化操作后,便将控制转到仙剑的main入口函数.仙剑程序调用库函数和Nanos-lite中自定义的库函数完成程序的运行.包括文件的读写和UI的显示等等.

实验内容

任务1：实现简单 loader

修改nanos-lite/src/loader.c实现loader.

```
void ramdisk_read(void *buf, off_t offset, size_t len);
void ramdisk_write(const void *buf, off_t offset, size_t len);
size_t get_ramdisk_size();

uintptr_t loader(_Protect *as, const char *filename) {
  ramdisk_read(DEFAULT_ENTRY, 0, get_ramdisk_size());
  return (uintptr_t)DEFAULT_ENTRY;
}
```

任务2.1：中断机制前的准备工作 && 任务2.2：实现中断机制

修改nemu/include/cpu/reg.h中的CPU_state联合.

```
typedef union {
  union {
    uint32_t _32;
    uint16_t _16;
    uint8_t _8[2];
  } gpr[8];

  struct {
    rtlreg_t eax;
    rtlreg_t ecx;
    rtlreg_t edx;
    rtlreg_t ebx;
    rtlreg_t esp;
    rtlreg_t ebp;
    rtlreg_t esi;
    rtlreg_t edi;
    vaddr_t eip;
    union {
      struct {
        uint32_t CF :1;
        uint32_t one:1;
        uint32_t :4;
        uint32_t ZF :1;
        uint32_t SF :1;
        uint32_t :1;
        uint32_t IF :1;
        uint32_t :1;
        uint32_t OF :1;
        uint32_t :20;
      } eflags;
      uint32_t eflags_num;
    };
    struct{
      uint16_t limit;
      uint32_t base;
    } idtr;
    uint16_t cs;
  };
} CPU_state;
```

修改nemu/src/cpu/exec/system.c实现lidt,int指令.

查表可知.

```
IF instruction = LIDT
THEN
  IF OperandSize = 16
  THEN
    IDTR.Limit:Base ← m16:24 (* 24 bits of base loaded *)
  ELSE
    IDTR.Limit:Base ← m16:32
  FI;
ELSE (* instruction = LGDT *)
  ...
FI;
```

```
void raise_intr(uint8_t NO, vaddr_t ret_addr);

make_EHelper(lidt) {
  cpu.idtr.limit = vaddr_read(id_dest->addr, 2);
  if (decoding.is_operand_size_16)
    cpu.idtr.base = vaddr_read(id_dest->addr + 2, 3);
  else
    cpu.idtr.base = vaddr_read(id_dest->addr + 2, 4);

  print_asm_template1(lidt);
}

...

make_EHelper(int) {
  raise_intr(id_dest->val, decoding.seq_eip);

  print_asm("int %s", id_dest->str);

#ifdef DIFF_TEST
  diff_test_skip_nemu();
#endif
}
```

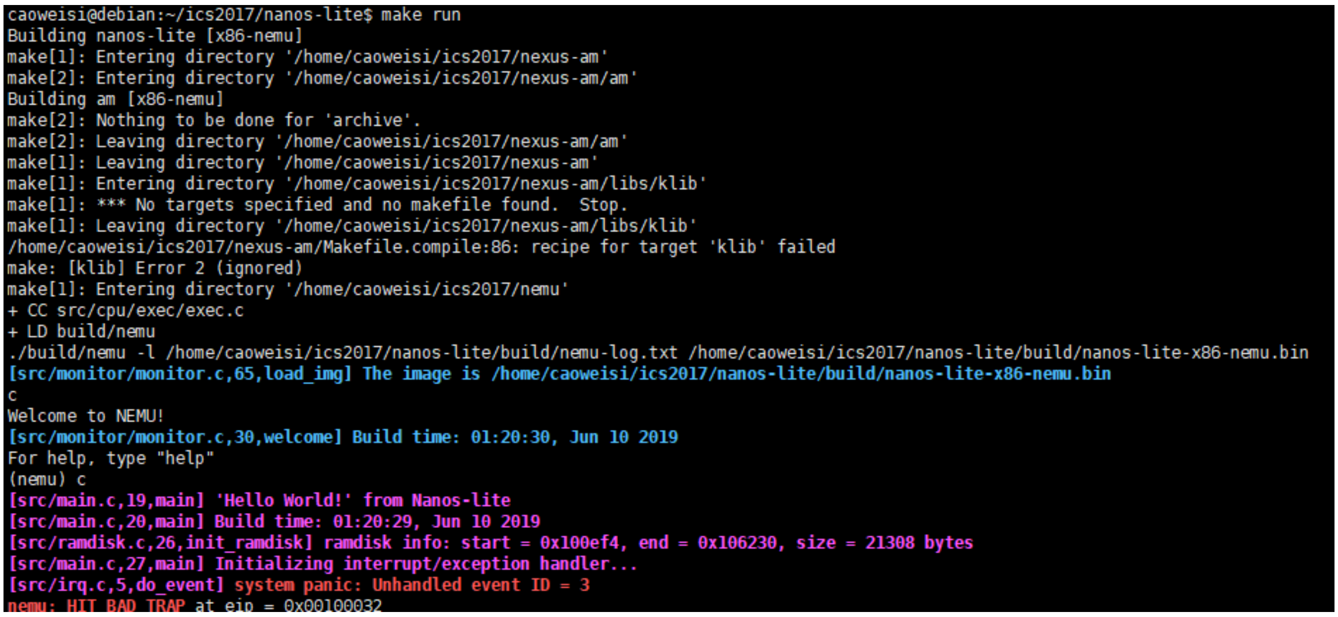
修改nemu/src/cpu/intr.c实现raise_intr().

```c
void raise_intr(uint8_t NO, vaddr_t ret_addr) {
  /* TODO: Trigger an interrupt/exception with ``NO''.
   * That is, use ``NO'' to index the IDT.
   */

  vaddr_t idt_addr;
  GateDesc gateDesc;

  rtl_push((rtlreg_t*)&cpu.eflags);
  rtl_push((rtlreg_t*)&cpu.cs);
  rtl_push((rtlreg_t*)&ret_addr);

  idt_addr = cpu.idtr.base + NO * 8;
  *(uint32_t *)&gateDesc = vaddr_read(idt_addr, 4);
  *((uint32_t *)&gateDesc + 1) = vaddr_read(idt_addr + 4, 4);

  decoding.is_jmp = 1;
  decoding.jmp_eip = (gateDesc.offset_31_16 << 16) | (gateDesc.offset_15_0 & 0xffff);
}
```

修改nemu/src/monitor/monitor.c的restart().

```c
static inline void restart() {
  /* Set the initial instruction pointer. */
  cpu.eip = ENTRY_START;
  cpu.eflags_num = 0x2;
  aou.cs = 8;

#ifdef DIFF_TEST
  init_qemu_reg();
#endif
}
```

运行成功.



```
caoweisi@debian:~/ics2017/nanos-lite$ make run
Building nanos-lite [x86-nemu]
make[1]: Entering directory '/home/caoweisi/ics2017/nexus-am'
make[2]: Entering directory '/home/caoweisi/ics2017/nexus-am/am'
Building am [x86-nemu]
make[2]: Nothing to be done for 'archive'.
make[2]: Leaving directory '/home/caoweisi/ics2017/nexus-am/am'
make[1]: Leaving directory '/home/caoweisi/ics2017/nexus-am'
make[1]: Entering directory '/home/caoweisi/ics2017/nexus-am/libs/klib'
make[1]: *** No targets specified and no makefile found.  Stop.
make[1]: Leaving directory '/home/caoweisi/ics2017/nexus-am/libs/klib'
/home/caoweisi/ics2017/nexus-am/Makefile.compile:86: recipe for target 'klib' failed
make: [klib] Error 2 (ignored)
make[1]: Entering directory '/home/caoweisi/ics2017/nemu'
+ CC src/cpu/exec/exec.c
+ LD build/nemu
./build/nemu -l /home/caoweisi/ics2017/nanos-lite/build/nemu-log.txt /home/caoweisi/ics2017/nanos-lite/build/nanos-lite-x86-nemu.bin
[src/monitor/monitor.c,65,load_img] The image is /home/caoweisi/ics2017/nanos-lite/build/nanos-lite-x86-nemu.bin
c
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 01:20:30, Jun 10 2019
For help, type "help"
(nemu) c
[src/main.c,19,main] 'Hello World!' from Nanos-lite
[src/main.c,20,main] Build time: 01:20:29, Jun 10 2019
[src/ramdisk.c,26,init_ramdisk] ramdisk info: start = 0x100ef4, end = 0x106230, size = 21308 bytes
[src/main.c,27,main] Initializing interrupt/exception handler...
[src/irq.c,5,do_event] system panic: Unhandled event ID = 3
nemu: HIT BAD TRAP at eip = 0x00100032
```

任务3：重新组织TrapFrame结构体

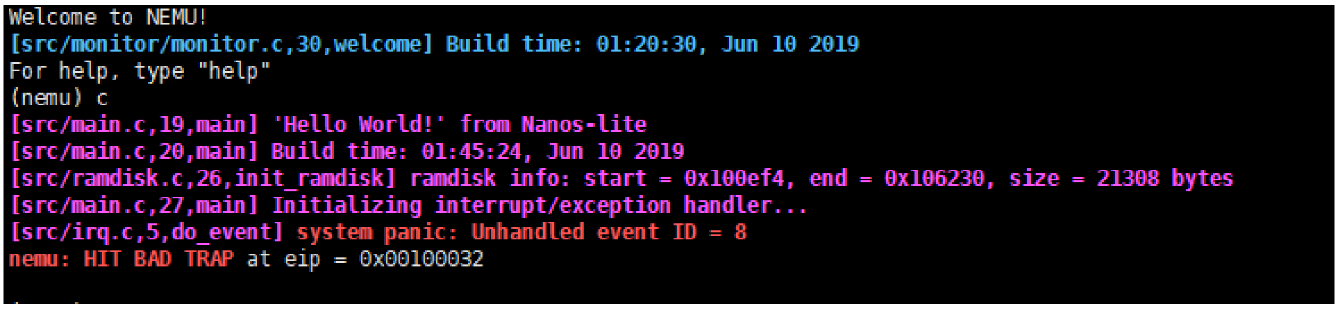
修改nemu/src/cpu/exec/data-mov.c实现pusha.

```
make_EHelper(pusha) {
  if (decoding.is_operand_size_16) {
    t0 = reg_w(R_SP);
    //保存当前sp
    rtl_push((rtlreg_t *)&reg_w(R_AX));
    rtl_push((rtlreg_t *)&reg_w(R_CX));
    rtl_push((rtlreg_t *)&reg_w(R_DX));
    rtl_push((rtlreg_t *)&reg_w(R_BX));
    rtl_push(&t0);
    rtl_push((rtlreg_t *)&reg_w(R_BP));
    rtl_push((rtlreg_t *)&reg_w(R_SI));
    rtl_push((rtlreg_t *)&reg_w(R_DI));
  }
  else {
    t0 = reg_w(R_ESP);
    //保存当前esp
    rtl_push((rtlreg_t *)&reg_w(R_EAX));
    rtl_push((rtlreg_t *)&reg_w(R_ECX));
    rtl_push((rtlreg_t *)&reg_w(R_EDX));
    rtl_push((rtlreg_t *)&reg_w(R_EBX));
    rtl_push(&t0);
    rtl_push((rtlreg_t *)&reg_w(R_EBP));
    rtl_push((rtlreg_t *)&reg_w(R_ESI));
    rtl_push((rtlreg_t *)&reg_w(R_EDI));
  }

  print_asm("pusha");
}
```

修改nexus-am/am/arch/x86-nemu/include/arch.h重定义_RegSet结构体.

```
struct _RegSet {
  uintptr_t edi, esi, ebp, esp, ebx, edx, ecx, eax;
  int irq;
  uintptr_t error, eip, cs, eflags;
};
```

运行成功.



```
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 01:20:30, Jun 10 2019
For help, type "help"
(nemu) c
[src/main.c,19,main] 'Hello World!' from Nanos-lite
[src/main.c,20,main] Build time: 01:45:24, Jun 10 2019
[src/ramdisk.c,26,init_ramdisk] ramdisk info: start = 0x100ef4, end = 0x106230, size = 21308 bytes
[src/main.c,27,main] Initializing interrupt/exception handler...
[src/irq.c,5,do_event] system panic: Unhandled event ID = 8
nemu: HIT BAD TRAP at eip = 0x00100032
```

任务4：实现系统调用

修改nanos-lite/src/irp.c中的do_event.

```
static _RegSet* do_event(_Event e, _RegSet* r) {
  switch (e.event) {
    case _EVENT_SYSCALL:
      return do_syscall(r);
    default: panic("Unhandled event ID = %d", e.event);
  }

  return NULL;
}
```

修改nexus-am/am/arch/x86-nemu/include/arch.h实现SYSCALL_ARGx()宏.

```
#define SYSCALL_ARG1(r) (r->eax)
#define SYSCALL_ARG2(r) (r->ebx)
#define SYSCALL_ARG3(r) (r->ecx)
#define SYSCALL_ARG4(r) (r->edx)
```

修改nanos-lite/src/syscall.c的do_syscall实现SYS_none和SYS_exit系统调用.

```
_RegSet* do_syscall(_RegSet *r) {
  uintptr_t a[4];
  a[0] = SYSCALL_ARG1(r);

  switch (a[0]) {
    case SYS_none:
      SYSCALL_ARG1(r) = 1;
      break;
    case SYS_exit:
      _halt(SYSCALL_ARG2(r));
      break;
    default: panic("Unhandled syscall ID = %d", a[0]);
  }

  return NULL;
}
```

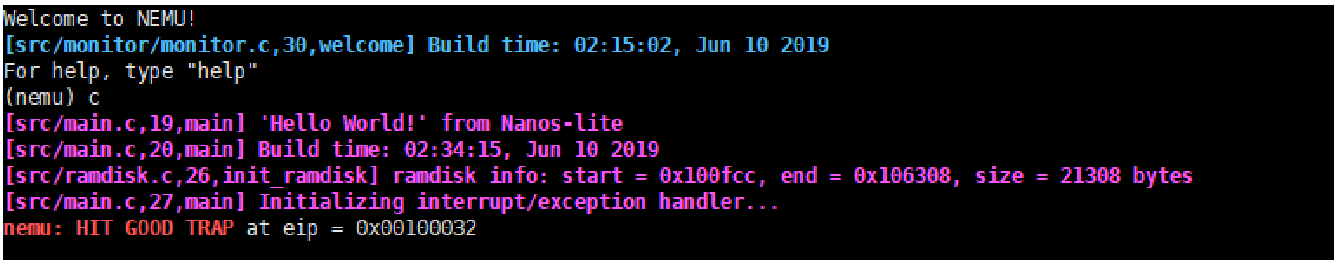
修改nemu/src/cpu/exec/data-mov.c实现popa.

```
make_EHelper(popa) {
  if (decoding.is_operand_size_16) {
    rtl_pop((rtlreg_t*)&reg_w(R_DI));
    rtl_pop((rtlreg_t*)&reg_w(R_SI));
    rtl_pop((rtlreg_t*)&reg_w(R_BP));
    rtl_pop(&t0);
    //pading
    rtl_pop((rtlreg_t*)&reg_w(R_BX));
    rtl_pop((rtlreg_t*)&reg_w(R_DX));
    rtl_pop((rtlreg_t*)&reg_w(R_CX));
    rtl_pop((rtlreg_t*)&reg_w(R_AX));
  }
  else {
    rtl_pop((rtlreg_t*)&reg_w(R_EDI));
    rtl_pop((rtlreg_t*)&reg_w(R_ESI));
    rtl_pop((rtlreg_t*)&reg_w(R_EBP));
    rtl_pop(&t0);
    //pading
    rtl_pop((rtlreg_t*)&reg_w(R_EBX));
    rtl_pop((rtlreg_t*)&reg_w(R_EDX));
    rtl_pop((rtlreg_t*)&reg_w(R_ECX));
    rtl_pop((rtlreg_t*)&reg_w(R_EAX));
  }
  print_asm("popa");
}
```

修改nemu/src/cpu/exec/system.c实现iret.

```
make_EHelper(iret) {
  rtl_pop(&t0);
  decoding.jmp_eip = t0;
  rtl_pop(&t0);
  cpu.cs = (uint16_t)t0;
  rtl_pop(&t0);
  cpu.eflags_num = t0;
  //恢复eip, cs, eflags.

  decoding.is_jmp = 1;

  print_asm("iret");
}
```

运行成功.



```
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 02:15:02, Jun 10 2019
For help, type "help"
(nemu) c
[src/main.c,19,main] 'Hello World!' from Nanos-lite
[src/main.c,20,main] Build time: 02:34:15, Jun 10 2019
[src/ramdisk.c,26,init_ramdisk] ramdisk info: start = 0x100fcc, end = 0x106308, size = 21308 bytes
[src/main.c,27,main] Initializing interrupt/exception handler...
nemu: HIT GOOD TRAP at eip = 0x00100032
```

任务5：在Nanos-lite上运行Hello world

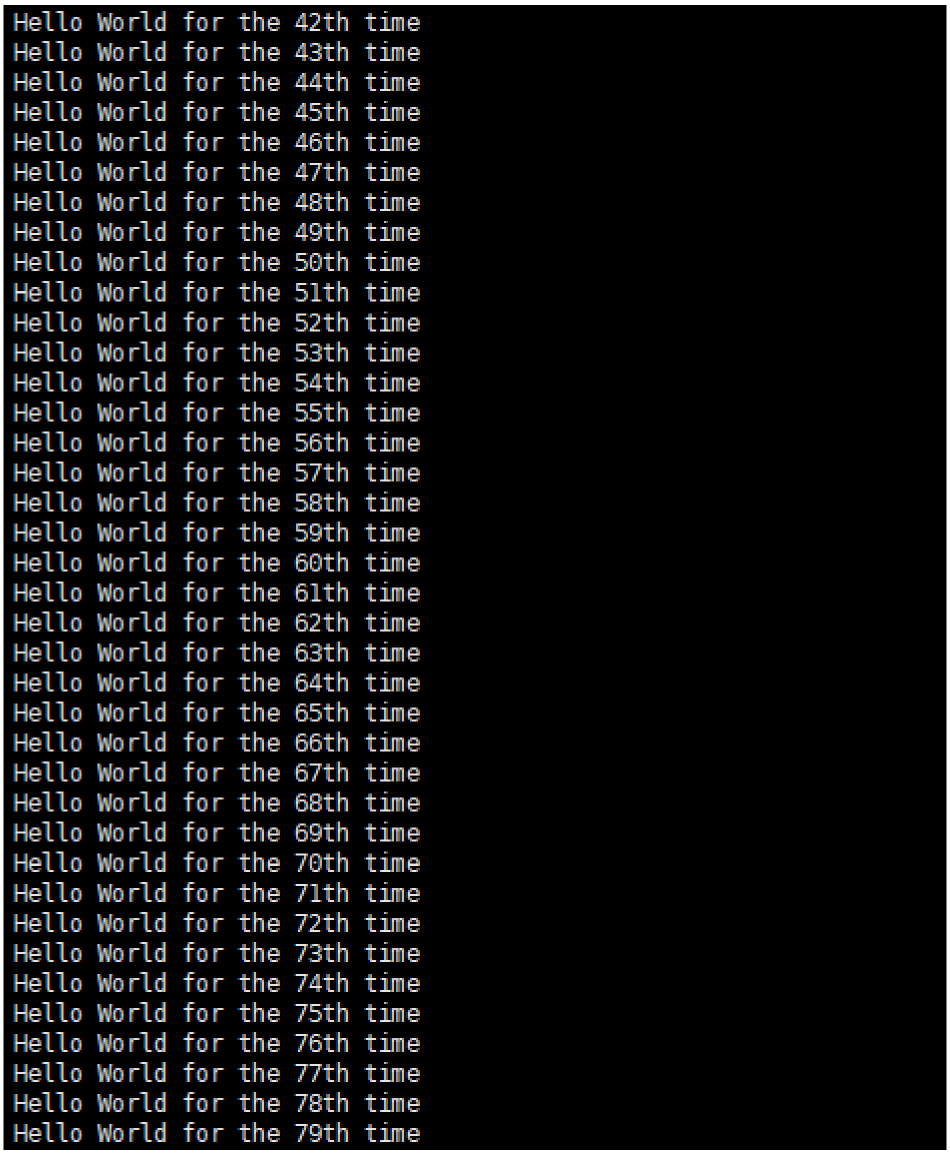
修改nanos-lite/src/syscall.c的do_syscall实现SYS_write系统调用.

```
static inline uintptr_t sys_write(uintptr_t fd, uintptr_t buf, uintptr_t len) {
  if (fd == 1 || fd == 2) {
    int i;
    for (i = 0; i < len; i++, buf++) {
      _putc(*(char*)buf);
    }
  }

  return 1;
}

_RegSet* do_syscall(_RegSet *r) {
  uintptr_t a[4];
  a[0] = SYSCALL_ARG1(r);

  switch (a[0]) {
    case SYS_none:
      SYSCALL_ARG1(r) = 1;
      break;
    case SYS_exit:
      _halt(SYSCALL_ARG2(r));
      break;
    case SYS_write:
      SYSCALL_ARG1(r) = sys_write(SYSCALL_ARG2(r), SYSCALL_ARG3(r), SYSCALL_ARG4(r));
      break;
    default: panic("Unhandled syscall ID = %d", a[0]);
  }

  return NULL;
}
```

修改navy-apps/libs/libos/src/nanos.c实现_write().

```
int _write(int fd, void *buf, size_t count){
  _syscall_(SYS_write, fd, (uintptr_t)buf, count);
}
```

运行成功.



```
Hello World for the 42th time
Hello World for the 43th time
Hello World for the 44th time
Hello World for the 45th time
Hello World for the 46th time
Hello World for the 47th time
Hello World for the 48th time
Hello World for the 49th time
Hello World for the 50th time
Hello World for the 51th time
Hello World for the 52th time
Hello World for the 53th time
Hello World for the 54th time
Hello World for the 55th time
Hello World for the 56th time
Hello World for the 57th time
Hello World for the 58th time
Hello World for the 59th time
Hello World for the 60th time
Hello World for the 61th time
Hello World for the 62th time
Hello World for the 63th time
Hello World for the 64th time
Hello World for the 65th time
Hello World for the 66th time
Hello World for the 67th time
Hello World for the 68th time
Hello World for the 69th time
Hello World for the 70th time
Hello World for the 71th time
Hello World for the 72th time
Hello World for the 73th time
Hello World for the 74th time
Hello World for the 75th time
Hello World for the 76th time
Hello World for the 77th time
Hello World for the 78th time
Hello World for the 79th time
```

任务6：实现堆区管理

修改nanos-lite/src/syscall.c的do_syscall实现SYS_brk系统调用.

```
static inline uintptr_t sys_brk(uintptr_t new_brk) {
  return (uintptr_t)mm_brk(new_brk);
}

_RegSet* do_syscall(_RegSet *r) {
  uintptr_t a[4];
  a[0] = SYSCALL_ARG1(r);

  switch (a[0]) {
    case SYS_none:
      SYSCALL_ARG1(r) = 1;
      break;
    case SYS_exit:
      _halt(SYSCALL_ARG2(r));
      break;
    case SYS_write:
      SYSCALL_ARG1(r) = sys_write(SYSCALL_ARG2(r), SYSCALL_ARG3(r), SYSCALL_ARG4(r));
      break;
    case SYS_brk:
      SYSCALL_ARG1(r) = sys_brk(SYSCALL_ARG2(r));
      break;
    default: panic("Unhandled syscall ID = %d", a[0]);
  }

  return NULL;
}
```

修改navy-apps/libs/libos/src/nanos.c实现_sbrk().

```
extern char _end;
//引用ld默认添加的符号
intptr_t program_break = (intptr_t)&_end;

...

void *_sbrk(intptr_t increment){
  intptr_t old_program_break = program_break;

  if (_syscall_(SYS_brk, program_break + increment, 0, 0) == 0) {
    program_break = program_break + increment;
    return (void *)old_program_break;
  }
  else {
    return -1;
  }
}
```

任务7：让loader使用文件

修改nanos-lite/src/loader.c实现loader.

```
uintptr_t loader(_Protect *as, const char *filename) {
  int fd = fs_open(filename, 0, 0);
  size_t f_size = fs_filesz(fd);
  fs_read(fd, DEFAULT_ENTRY, f_size);
  fs_close(fd);

  return (uintptr_t)DEFAULT_ENTRY;
}
```

任务：实现完整的文件系统 && 任务8：实现系统调用

修改nanos-lite/src/syscall.c的do_syscall实现相应系统调用.

```c
static inline uintptr_t sys_open(uintptr_t pathname, uintptr_t flags, uintptr_t mode) {
  return fs_open((char *)pathname, flags, mode);
}

static inline uintptr_t sys_read(uintptr_t fd, uintptr_t buf, uintptr_t len) {
  return fs_read(fd, (void*)buf, len);
}

static inline uintptr_t sys_lseek(uintptr_t fd, uintptr_t offset, uintptr_t whence) {
  return fs_lseek(fd, offset, whence);
}

static inline uintptr_t sys_close(uintptr_t fd) {
  return fs_close(fd);
}

static inline uintptr_t sys_brk(uintptr_t new_brk) {
  return (uintptr_t)mm_brk(new_brk);
}

static inline uintptr_t sys_write(uintptr_t fd, uintptr_t buf, uintptr_t len) {
  return fs_write(fd, (void*)buf, len);
}

_RegSet* do_syscall(_RegSet *r) {
  uintptr_t a[4];
  a[0] = SYSCALL_ARG1(r);

  switch (a[0]) {
    case SYS_none:
      SYSCALL_ARG1(r) = 1;
      break;
    case SYS_exit:
      _halt(SYSCALL_ARG2(r));
      break;
    case SYS_write:
      SYSCALL_ARG1(r) = sys_write(SYSCALL_ARG2(r), SYSCALL_ARG3(r), SYSCALL_ARG4(r));
      break;
    case SYS_open:
      SYSCALL_ARG1(r) = sys_open(SYSCALL_ARG2(r), SYSCALL_ARG3(r), SYSCALL_ARG4(r));
      break;
    case SYS_read:
      SYSCALL_ARG1(r) = sys_read(SYSCALL_ARG2(r), SYSCALL_ARG3(r), SYSCALL_ARG4(r));
      break;
    case SYS_lseek:
      SYSCALL_ARG1(r) = sys_lseek(SYSCALL_ARG2(r), SYSCALL_ARG3(r), SYSCALL_ARG4(r));
      break;
    case SYS_close:
      SYSCALL_ARG1(r) = sys_close(SYSCALL_ARG2(r));
      break;
    case SYS_brk:
      SYSCALL_ARG1(r) = sys_brk(SYSCALL_ARG2(r));
      break;
    default: panic("Unhandled syscall ID = %d", a[0]);
  }

  return NULL;
}
```

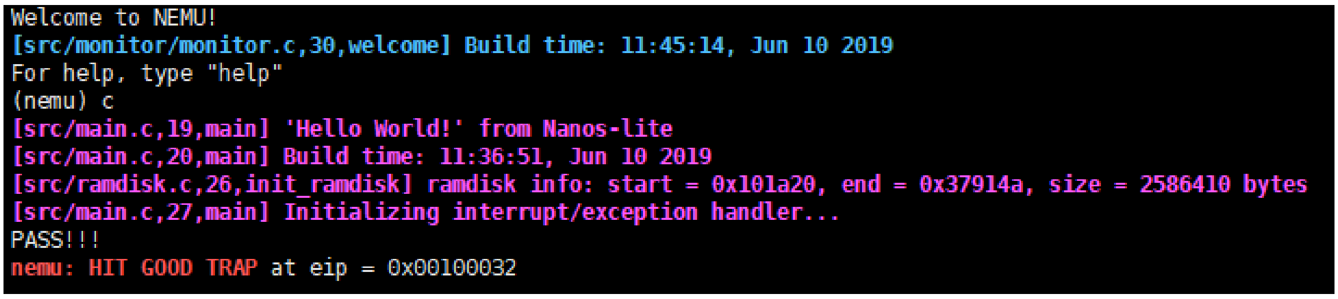
修改navy-apps/libs/libos/src/nanos.c中的相应接口函数.

```
int _open(const char *path, int flags, mode_t mode) {
  return _syscall_(SYS_open, (uintptr_t)path, flags, mode);
}

int _write(int fd, void *buf, size_t count){
  return _syscall_(SYS_write, fd, (uintptr_t)buf, count);
}

int _read(int fd, void *buf, size_t count) {
  return _syscall_(SYS_read, fd, (uintptr_t)buf, count);
}

int _close(int fd) {
  return _syscall_(SYS_close, fd, 0, 0);
}

off_t _lseek(int fd, off_t offset, int whence) {
  return _syscall_(SYS_lseek, fd, offset, whence);
}
```

修改nanos-lite\src\main.c中loader函数的参数.

```
uint32_t entry = loader(NULL, "/bin/text");
```

运行成功.



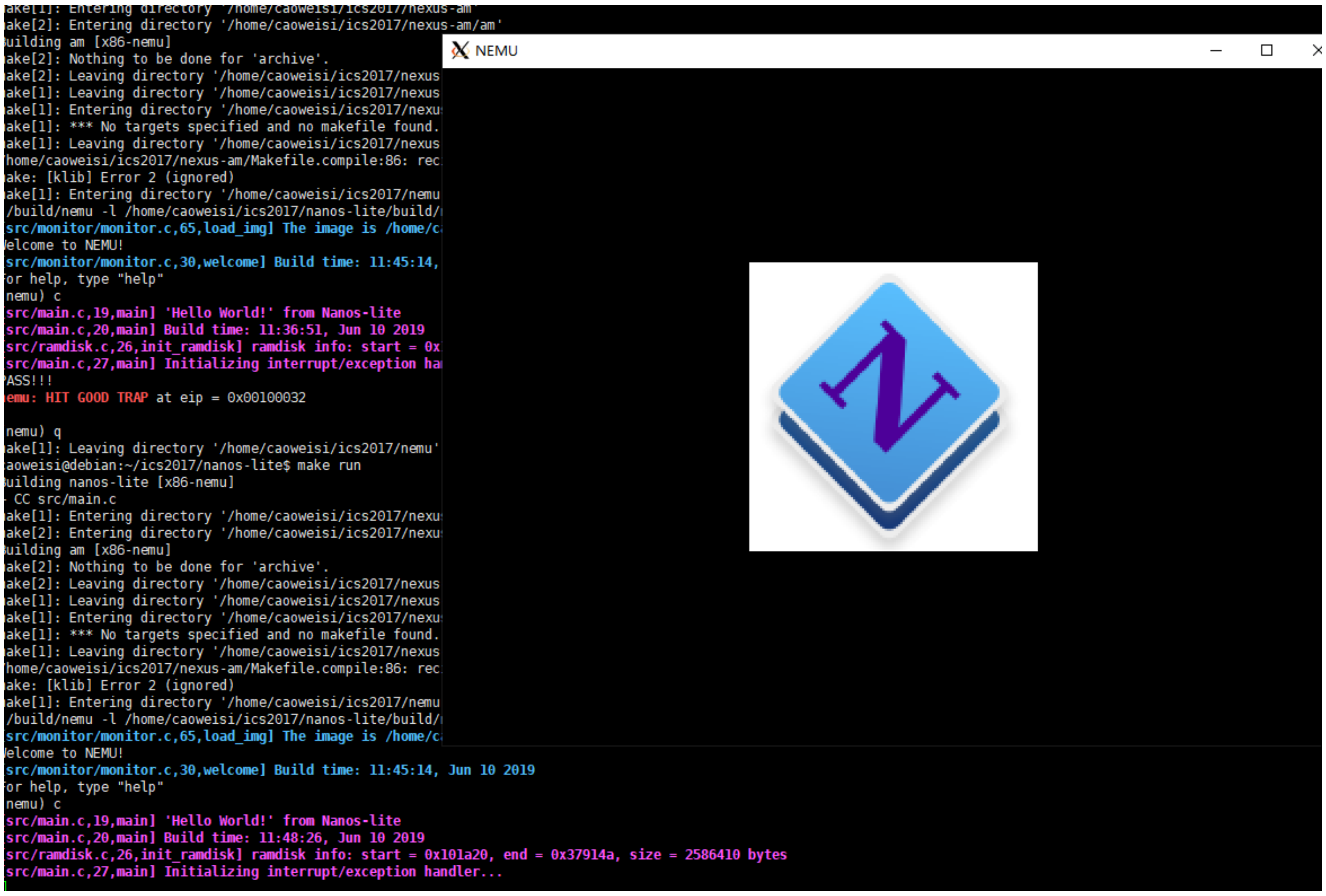
```
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 11:45:14, Jun 10 2019
For help, type "help"
(nemu) c
[src/main.c,19,main] 'Hello World!' from Nanos-lite
[src/main.c,20,main] Build time: 11:36:51, Jun 10 2019
[src/ramdisk.c,26,init_ramdisk] ramdisk info: start = 0x101a20, end = 0x37914a, size = 2586410 bytes
[src/main.c,27,main] Initializing interrupt/exception handler...
PASS!!!
nemu: HIT GOOD TRAP at eip = 0x00100032
```

任务9：把VGA显存抽象成文件

修改nanos-lite\src\main.c中loader函数的参数.

```
uint32_t entry = loader(NULL, "/bin/bmptest");
```

运行成功.

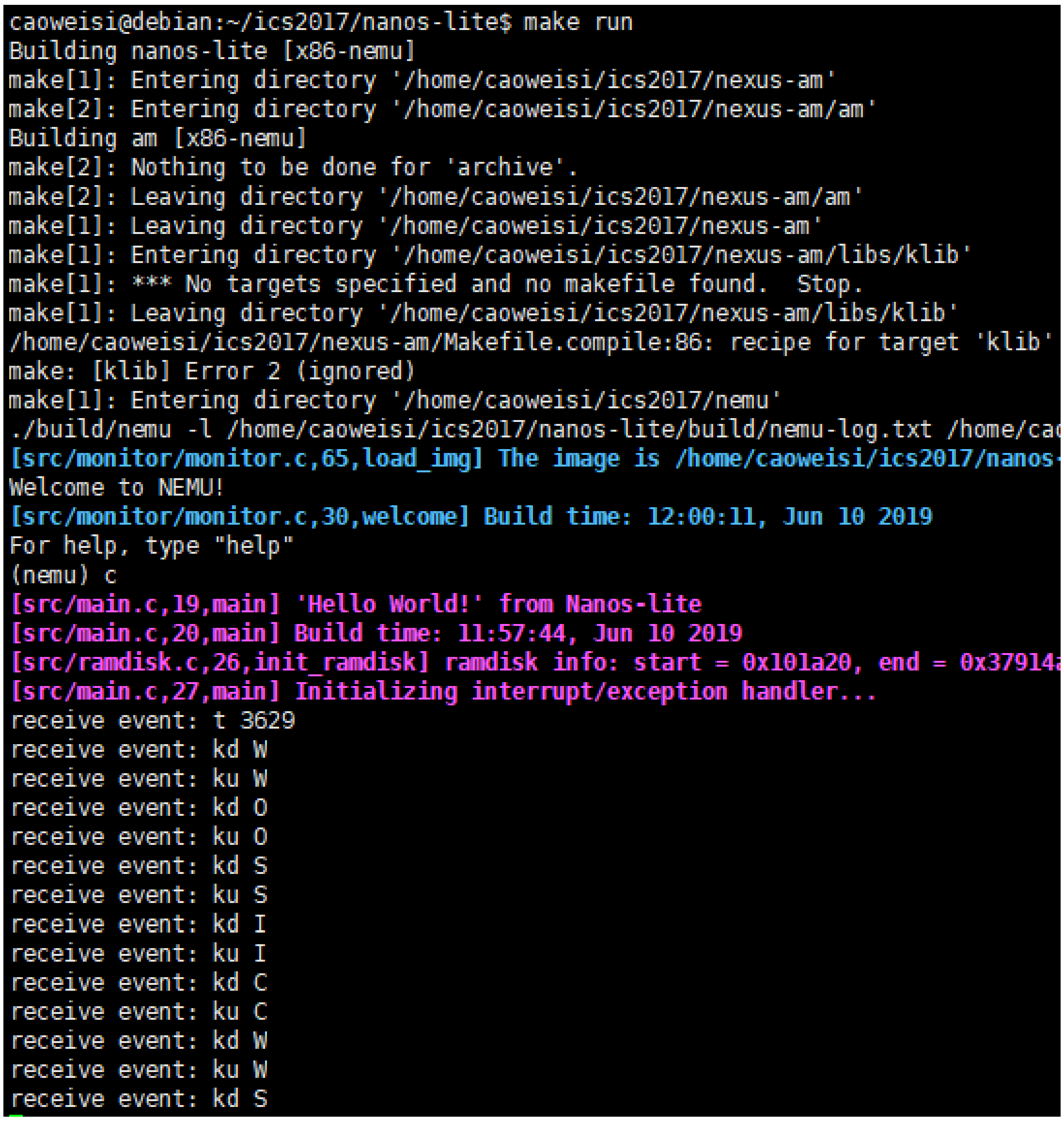

```
ake[1]: Entering directory '/home/caoweisi/ics2017/nexus-am'
ake[2]: Entering directory '/home/caoweisi/ics2017/nexus-am/am'
uilding am [x86-nemu]
ake[2]: Nothing to be done for 'archive'.
ake[2]: Leaving directory '/home/caoweisi/ics2017/nexus
ake[1]: Leaving directory '/home/caoweisi/ics2017/nexus
ake[1]: Entering directory '/home/caoweisi/ics2017/nexu
ake[1]: *** No targets specified and no makefile found.
ake[1]: Leaving directory '/home/caoweisi/ics2017/nexus
home/caoweisi/ics2017/nexus-am/Makefile.compile:86: rec
ake: [klib] Error 2 (ignored)
ake[1]: Entering directory '/home/caoweisi/ics2017/nemu
/build/nemu -l /home/caoweisi/ics2017/nanos-lite/build/
src/monitor/monitor.c,65,load_img] The image is /home/c
elcome to NEMU!
src/monitor/monitor.c,30,welcome] Build time: 11:45:14,
or help, type "help"
nemu) c
src/main.c,19,main] 'Hello World!' from Nanos-lite
src/main.c,20,main] Build time: 11:36:51, Jun 10 2019
src/ramdisk.c,26,init_ramdisk] ramdisk info: start = 0x
src/main.c,27,main] Initializing interrupt/exception ha
ASS!!!
emu: HIT GOOD TRAP at eip = 0x00100032

nemu) q
ake[1]: Leaving directory '/home/caoweisi/ics2017/nemu'
aoweisi@debian:~/ics2017/nanos-lite$ make run
uilding nanos-lite [x86-nemu]
 CC src/main.c
ake[1]: Entering directory '/home/caoweisi/ics2017/nexu
ake[2]: Entering directory '/home/caoweisi/ics2017/nexu
uilding am [x86-nemu]
ake[2]: Nothing to be done for 'archive'.
ake[2]: Leaving directory '/home/caoweisi/ics2017/nexus
ake[1]: Leaving directory '/home/caoweisi/ics2017/nexus
ake[1]: Entering directory '/home/caoweisi/ics2017/nexu
ake[1]: *** No targets specified and no makefile found.
ake[1]: Leaving directory '/home/caoweisi/ics2017/nexus
home/caoweisi/ics2017/nexus-am/Makefile.compile:86: rec
ake: [klib] Error 2 (ignored)
ake[1]: Entering directory '/home/caoweisi/ics2017/nemu
/build/nemu -l /home/caoweisi/ics2017/nanos-lite/build/
src/monitor/monitor.c,65,load_img] The image is /home/c
elcome to NEMU!
src/monitor/monitor.c,30,welcome] Build time: 11:45:14, Jun 10 2019
or help, type "help"
nemu) c
src/main.c,19,main] 'Hello World!' from Nanos-lite
src/main.c,20,main] Build time: 11:48:26, Jun 10 2019
src/ramdisk.c,26,init_ramdisk] ramdisk info: start = 0x101a20, end = 0x37914a, size = 2586410 bytes
src/main.c,27,main] Initializing interrupt/exception handler...
```

任务10：把设备输入抽象成文件

修改nanos-lite\src\main.c中loader函数的参数.

```
uint32_t entry = loader(NULL, "/bin/events");
```

运行成功.



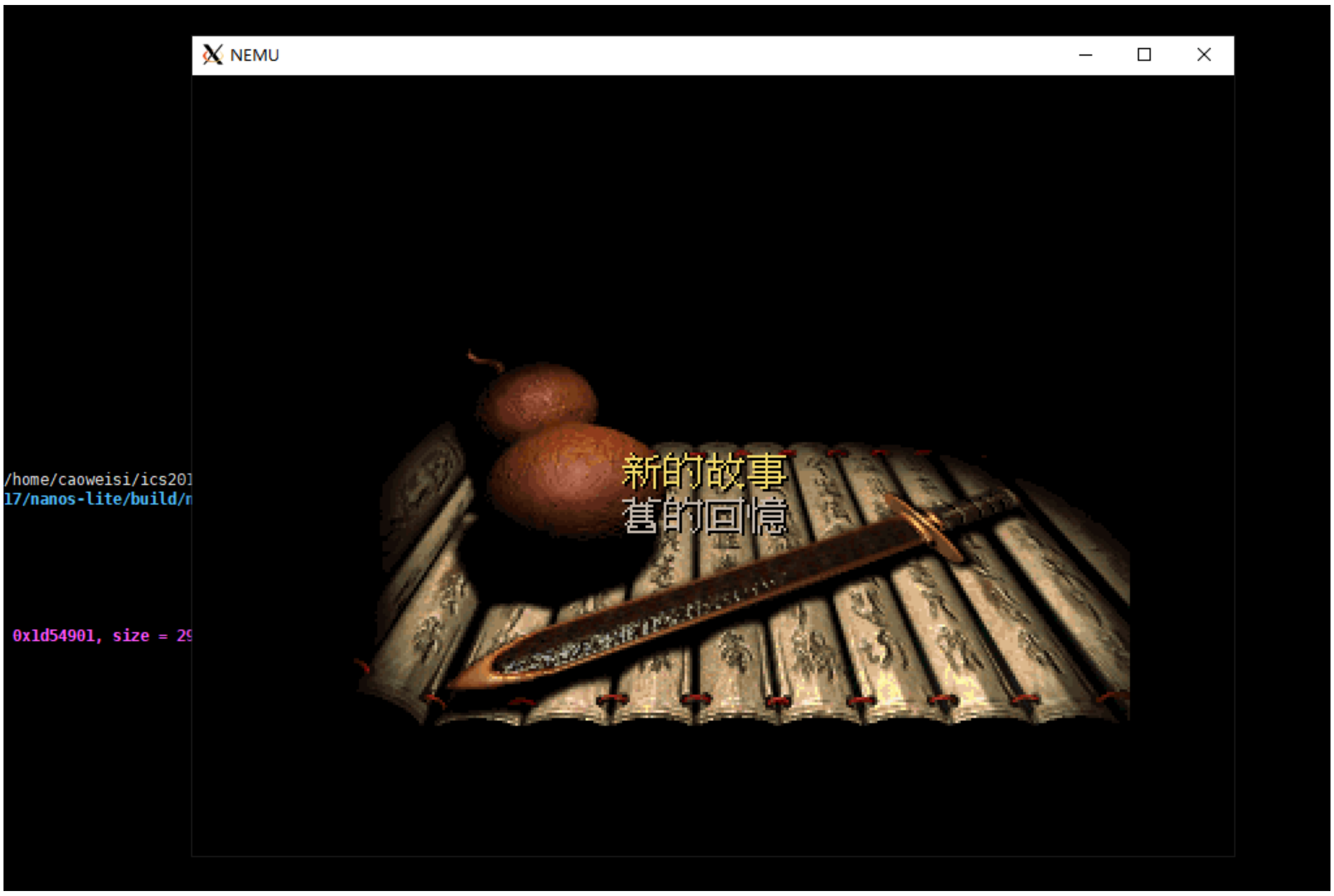
```
caoweisi@debian:~/ics2017/nanos-lite$ make run
Building nanos-lite [x86-nemu]
make[1]: Entering directory '/home/caoweisi/ics2017/nexus-am'
make[2]: Entering directory '/home/caoweisi/ics2017/nexus-am/am'
Building am [x86-nemu]
make[2]: Nothing to be done for 'archive'.
make[2]: Leaving directory '/home/caoweisi/ics2017/nexus-am/am'
make[1]: Leaving directory '/home/caoweisi/ics2017/nexus-am'
make[1]: Entering directory '/home/caoweisi/ics2017/nexus-am/libs/klib'
make[1]: *** No targets specified and no makefile found.  Stop.
make[1]: Leaving directory '/home/caoweisi/ics2017/nexus-am/libs/klib'
/home/caoweisi/ics2017/nexus-am/Makefile.compile:86: recipe for target 'klib'
make: [klib] Error 2 (ignored)
make[1]: Entering directory '/home/caoweisi/ics2017/nemu'
./build/nemu -l /home/caoweisi/ics2017/nanos-lite/build/nemu-log.txt /home/cao
[src/monitor/monitor.c,65,load_img] The image is /home/caoweisi/ics2017/nanos
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 12:00:11, Jun 10 2019
For help, type "help"
(nemu) c
[src/main.c,19,main] 'Hello World!' from Nanos-lite
[src/main.c,20,main] Build time: 11:57:44, Jun 10 2019
[src/ramdisk.c,26,init_ramdisk] ramdisk info: start = 0x101a20, end = 0x37914a
[src/main.c,27,main] Initializing interrupt/exception handler...
receive event: t 3629
receive event: kd W
receive event: ku W
receive event: kd O
receive event: ku O
receive event: kd S
receive event: ku S
receive event: kd I
receive event: ku I
receive event: kd C
receive event: ku C
receive event: kd W
receive event: ku W
receive event: kd S
```

任务11：在NEMU中运行仙剑奇侠传

修改nanos-lite\src\main.c中loader函数的参数.

```
uint32_t entry = loader(NULL, "/bin/pal");
```

运行成功.



```
/home/caoweisi/ics201
17/nanos-lite/build/n

 0x1d54901, size = 29
```

遇到的问题及解决办法

拉取框架代码时发现对之前修改的代码产生了覆盖,很奇怪为什么没有问merge,一开始没发现一直卡在这里.

实验心得

无.

其他备注

无.