南京航空航天大学 计算机科学与技术系学院 计算机组成原理 课程实验

学号：161630220

姓名：赵维康

PA2- 简单复杂的机器: 冯诺依曼计算机系统

在进行本 PA 前， 请在工程目录下执行以下命令进行分支整理，否则将影响你的成绩:
git commit --allow-empty -am "before starting pa2"
git checkout master
git merge pa1
git checkout -b pa2

以下是代码执行截图

```
zhaoweikang@zhaoweikang:~/ics2017$ sudo git commit --allow-empty -am "before sta
rting pa2"
[sudo] zhaoweikang 的密码：
[pa1 cc11412] before starting pa2
zhaoweikang@zhaoweikang:~/ics2017$ sudo git checkout master
切换到分支 'master'
您的分支领先 'origin/2017' 共 9 个提交。
  （使用 "git push" 来发布您的本地提交）
zhaoweikang@zhaoweikang:~/ics2017$ sudo git merge pa1
更新 a7511bf..cc11412
Fast-forward
 nemu/include/monitor/watchpoint.h   |  10 +-
 nemu/src/monitor/debug/expr.c       | 292 ++++++++++++++++++++++++++++++++++---
 nemu/src/monitor/debug/ui.c         | 114 +++++++++++++++-
 nemu/src/monitor/debug/watchpoint.c |  82 +++++++++++-
 4 files changed, 467 insertions(+), 31 deletions(-)
zhaoweikang@zhaoweikang:~/ics2017$ sudo git checkout -b pa2
切换到一个新分支 'pa2'
```

运行第一个 C 程序
在 nexus-am/tests/cputest 目录下键入 make ARCH=x86-nemu ALL=dummy run
编译 dummy 程序,并启动 NEMU,截图如图

```
root@zhaoweikang:/home/zhaoweikang/ics2017/nexus-am/tests/cputest# make ARCH=x86
-nemu ALL=dummy run
Building dummy [x86-nemu]
+ CC tests/dummy.c
Building am [x86-nemu]
+ CC arch/x86-nemu/src/trm.c
+ AS arch/x86-nemu/src/trap.S
+ CC arch/x86-nemu/src/pte.c
+ CC arch/x86-nemu/src/ioe.c
+ CC arch/x86-nemu/src/asye.c
+ AR /home/zhaoweikang/ics2017/nexus-am/am/build/am-x86-nemu.a
make[2]: *** 没有指明目标并且找不到 makefile。 停止。
+ CC src/monitor/cpu-exec.c
+ CC src/monitor/debug/expr.c
+ CC src/monitor/debug/watchpoint.c
+ CC src/monitor/debug/ui.c
+ CC src/cpu/exec/exec.c
fatal: ..: '..' 在仓库之外
+ LD build/nemu
fatal: ..: '..' 在仓库之外
[src/monitor/monitor.c,65,load_img] The image is /home/zhaoweikang/ics2017/nexus
-am/tests/cputest/build/dummy-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 12:38:57, Mar 14 2018
For help, type "help"
(nemu) c
invalid opcode(eip = 0x0010000a): e8 01 00 00 00 90 55 89 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x0010000a is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x0010000a) in the disassembling result to distinguish which case
it is.

If it is the first case, see
 _ ____   ___    __    __ __                          _
(_)___ \ / _ \  / /   |  \/  |                        | |
 _  __) | (_) |/ /_   | \  / | __ _ _ __   _   _  __ _| |
| ||__ < > _ <| '_ \  | |\/| |/ _` | '_ \| | | |/ _` | |
| |___) | (_) | (_) | | |  | | (_| | | | | |_| | (_| | |
|_|____/ \___/ \___/  |_|  |_|\__,_|_| |_|\__,_|\__,_|_|

for more details.

If it is the second case, remember:
* The machine is always right!
* Every line of untested code is always wrong!
```

下面进入 **nexus-am/tests/cputest/build/dummy-x86-nemu.txt** 中，查看反汇编结果,如图

```
/home/zhaoweikang/ics2017/nexus-am/tests/cputest/build/dummy-x86-nemu :     文件格式 elf32-i386


Disassembly of section .text:

00100000 <_start>:
  100000:       bd 00 00 00 00          mov    $0x0,%ebp
  100005:       bc 00 7c 00 00          mov    $0x7c00,%esp
  10000a:       e8 01 00 00 00          call   100010 <_trm_init>
  10000f:       90                      nop

00100010 <_trm_init>:
  100010:       55                      push   %ebp
  100011:       89 e5                   mov    %esp,%ebp
  100013:       83 ec 08                sub    $0x8,%esp
  100016:       e8 05 00 00 00          call   100020 <main>
  10001b:       d6                      (bad)
  10001c:       eb fe                   jmp    10001c <_trm_init+0xc>
  10001e:       66 90                   xchg   %ax,%ax

00100020 <main>:
  100020:       55                      push   %ebp
  100021:       89 e5                   mov    %esp,%ebp
  100023:       31 c0                   xor    %eax,%eax
  100025:       5d                      pop    %ebp
  100026:       c3                      ret
```

下面进入 **nemu/include/cpu/reg.h** 文件编写 **eflags** 寄存器,其中用到了 **union** 联合体与 **struct** 结构体，同时也用到了 **eflags** 寄存器位域的概念，代码如图

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

  /* Do NOT change the order of the GPRs' definitions. */

  /* In NEMU, rtlreg_t is exactly uint32_t. This makes RTL instructions
   * in PA2 able to directly access these registers.
   */
  struct{
      rtlreg_t eax, ecx, edx, ebx, esp, ebp, esi, edi;
        };
      };
      vaddr_t eip;
      unsigned int cs;
       union{
             rtlreg_t eflags_init;
             struct{
                  unsigned int CF:1;
                  unsigned int ZF:1;
                  unsigned int SF:1;
                  unsigned int IF:1;
                  unsigned int OF:1;
                    };
                  };
                  }eflags;

} CPU_state;
```

下面进入 **nemu/include/cpu/rtl.h** 文件，实现一些基本的 **RTL** 指令，包括 **EFLAGS** 标志位的

读写即 rtl_set_(CF|OF|ZF|SF|IF) 和 rtl_get_(CF|OF|ZF|SF|IF)；rtl_push()和 rtl_pop()；数据移动 rtl_mv；符号扩展 rtl_sext；rtl_not；以及 eflags 的更新，即 rtl_update_ZF()和 rtl_update_SF()。以下是实现截图

```
#define make_rtl_setget_eflags(f) \
  static inline void concat(rtl_set_, f) (const rtlreg_t* src) { \
   cpu.eflags.f=*src; \
  } \
  static inline void concat(rtl_get_, f) (rtlreg_t* dest) { \
   *dest=cpu.eflags.f; \
  }
```

```
static inline void rtl_push(const * src1) {
  // esp <- esp - 4
  // M[esp] <- src1
  cpu.esp -= 4;
  vaddr_write(cpu.esp,4,*src1);
}
```

```
static inline void rtl_pop(rtlreg_t* dest) {
  // dest <- M[esp]
  // esp <- esp + 4
  *dest=vaddr_read(cpu.esp,4);
  cpu.esp += 4;
}
```

```
static inline void rtl_sext(rtlreg_t* dest, const rtlreg_t* src1, int width) {
  // dest <- signext(src1[(width * 8 - 1) .. 0])
  switch(width){
         case 1:
                *dest=(int32_t)(int16_t) *src1;
                return ;
         case 2:
                *dest=(int32_t)(int16_t) *src1;
                return ;
         case 4:
                *dest=(int32_t) *src1;
                return ;
         }
}
```

```
static inline void rtl_mv(rtlreg_t* dest, const rtlreg_t *src1) {
  // dest <- src1
  *dest=*src1;
}
```

```
static inline void rtl_not(rtlreg_t* dest) {
  // dest <- ~dest
  *dest=~(*dest);
}
```

```
static inline void rtl_update_ZF(const rtlreg_t* result, int width) {
  // eflags.ZF <- is_zero(result[width * 8 - 1 .. 0])
  int zf = 0;
  if(width == 1){
          zf =(*result & 0x000000ff) | 0;
  }
  else if(width == 2){
          zf =(*result & 0x0000ffff) | 0;
  }
  else if(width == 4){
          zf =(*result & 0xffffffff) | 0;
  }
  cpu.eflags.ZF =(zf == 0) ? 1 : 0;
}
```

```
static inline void rtl_update_SF(const rtlreg_t* result, int width) {
  // eflags.SF <- is_sign(result[width * 8 - 1 .. 0])
  int sf = 0;
  sf = (*result >> (width * 8 - 1)) & 0x1;
  cpu.eflags.SF = sf;
}
```

在实现具体相关指令的执行函数之前，先进入 **nemu/src/cpu/exec/all-instr.h** 文件，对要实现的每个指令的执行函数进行声明，以免后面运行 **dummy** 时出现函数未声明的错误，以下是具体实现

```
#include "cpu/exec.h"

make_EHelper(mov);


make_EHelper(call);

make_EHelper(ret);


make_EHelper(push);

make_EHelper(pop);


make_EHelper(sub);


make_EHelper(xor);


make_EHelper(operand_size);


make_EHelper(inv);
make_EHelper(nemu_trap);
```

然后 **make** 以及 **make run** 一下，如图

```
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make
+ CC src/cpu/exec/exec.c

+ LD build/nemu
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make run
```

```
./build/nemu -l ./build/nemu-log.txt
[src/monitor/monitor.c,47,load_default_img] No image is given. Use the default b
uild-in image.
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 10:42:15, Apr 15 2018
For help, type "help"
(nemu)
```

说明实现没有错误

下面进入 nemu/src/cpu/exec/control.c 文件中，实现 call 指令以及 ret 指令，以下是具体实现截图

```
make_EHelper(call) {
  // the target address is calculated at the decode stage
  rtl_push(& decoding.sep_eip);
  decoding.is_jmp = 1;
  print_asm("call %x", decoding.jmp_eip);
}
```

call：读取要压栈的 eip 值。该 eip 值其实就是 call 的下一条指令的首地址，该 eip 值保存在全局变量 decoding.seq_eip 中，接着调用 rtl_push 将&decoding.seq_eip 压栈，然后设置：decoding.is_jmp = 1，并将 decoding.jmp_eip 设为跳转目标地址。

```
make_EHelper(ret) {
  rtl_pop(& decoding.jmp_eip);
  decoding.is_jmp = 1;
  print_asm("ret");
}
```

ret：调用 rtl_pop 取栈顶元素 eip 的值，也即&decoding.jmp_eip，然后设置跳转 decoding.is_jmp = 1，调用 print_asm()打印 ret。

然后 make 以及 make run 一下，如图

```
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make
+ CC src/cpu/exec/control.c
+ LD build/nemu
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make run
./build/nemu -l ./build/nemu-log.txt
[src/monitor/monitor.c,47,load_default_img] No image is given. Use the default b
uild-in image.
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 10:42:15, Apr 15 2018
For help, type "help"
(nemu)
```

说明实现没有错误

下面进入 nemu/src/exec/data-mov.c 文件，实现 push 指令以及 pop 指令的执行函数，以下是具体实现

```
make_EHelper(push) {
  if(id_dest->width == 1){
            id_dest->val = (int32_t)(int8_t) id_dest->val;
  }
  rtl_push(& id_dest->val);
  print_asm_template1(push);
}
```

push：调用使用 RTL 实现的 rtl_push(),将保存在 id_dest 中的值即 id_dest->val，同时进行访存操作，调用 print_asm_template1()打印单目操作数指令 push 的反汇编结果

```
make_EHelper(pop) {
  rtl_pop(& t0);
  operand_write(id_dest,&t0);
  print_asm_template1(pop);
}
```

pop：调用使用 RTL 实现的 rtl_pop()，将要出栈的栈顶元素保存在临时寄存器 t0 中，然后掉用 operand_write()进行写回，调用 print_asm_template1()打印单目操作数指令 pop 的反汇编结果

然后 make 以及 make run 一下，如图

```
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make
[sudo] zhaoweikang 的密码：
+ CC src/cpu/exec/data-mov.c

+ LD build/nemu
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make run
./build/nemu -l ./build/nemu-log.txt
[src/monitor/monitor.c,47,load_default_img] No image is given. Use the default b
uild-in image.
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 10:42:15, Apr 15 2018
For help, type "help"
(nemu)
```

说明实现没有错误

下面进入 nemu/stc/exec/arith.c 文件，完成对 sub 指令执行函数的编写，下面是具体实现

```
make_EHelper(sub) {
  rtl_sub(&t2,&id_dest->val,&id_src->val);
        operand_write(id_dest,&t2);

  rtl_update_ZFSF(&t2,id_dest->width);

  rtl_sltu(&t0,&id_dest->val,&t2);
  rtl_set_CF(&t0);

  rtl_xor(&t0,&id_dest->val,&id_src->val);
  rtl_xor(&t1,&id_dest->val,&t2);
  rtl_and(&t0,&t0,&t1);
  rtl_msb(&t0,&t0,id_dest->width);
  rtl_set_OF(&t0);

  print_asm_template2(sub);
}
```

然后 make 以及 make run 一下，如图

```
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make
+ CC src/cpu/exec/arith.c

+ LD build/nemu
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make run
```

```
./build/nemu -l ./build/nemu-log.txt
[src/monitor/monitor.c,47,load_default_img] No image is given. Use the default b
uild-in image.
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 10:42:15, Apr 15 2018
For help, type "help"
(nemu) q
```

说明实现没有错误

sub：仿照 **sbb** 指令的执行函数的实现来完成，做些个别改动就可以。
同时，进入 **nemu/src/monitor/monitor.c** 文件，在 **restart()** 函数中对 **EFLAGS** 寄存器进行初始化，如图

```
static inline void restart() {
  /* Set the initial instruction pointer. */
  cpu.eip = ENTRY_START;
        cpu.cs = 0x8;
        cpu.eflags.eflags_init =0x2;

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

下面进入 **nemu/src/exec/logic.c** 文件，进行 **xor** 指令执行函数的实现，下面是实现代码

```
make_EHelper(xor) {
  rtl_xor(&t2,&id_dest->val,&id_src->val);
        operand_write(id_dest,&t2);
        rtl_update_ZFSF(&t2,id_dest->width);
        rtl_set_CF(&tzero);
        rtl_set_OF(&tzero);

  print_asm_template2(xor);
}
```

然后 **make** 以及 **make run** 一下，如图

```
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make
+ CC src/cpu/exec/arith.c
+ CC src/cpu/exec/logic.c

+ LD build/nemu
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo make run
./build/nemu -l ./build/nemu-log.txt
[src/monitor/monitor.c,47,load_default_img] No image is given. Use the default b
uild-in image.
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 10:42:15, Apr 15 2018
For help, type "help"
(nemu)
```

说明实现没有错误

xor：调用 **RTL** 指令 **rtl_xor()**进行访存，然后 **operand_write** 写回，同时更新 **ZFSF** 标志位,以及设置 **CF**、**OF** 标志位，最后调用 **print_asm_template2()**打印双目操作数指令 **xor** 的反汇编代码。

进入 nemu/src/cpu/exec/exec.c 文件 ， 下面填写 opcode_table[]表，具体实现如

call:

```
/* 0xe8 */    IDEX(J, call), EMPTY, EMPTY, EMPTY,
```

push:

```
 /* 0x50 */    IDEX(r, push), IDEX(r, push), IDEX(r, push), IDEX(r, push),

 /* 0x54 */    IDEX(r, push), IDEX(r, push), IDEX(r, push), IDEX(r, push),
```

pop:

```
 /* 0x58 */    IDEX(r, pop), IDEX(r, pop), IDEX(r, pop), IDEX(r, pop),
 /* 0x5c */    IDEX(r, pop), IDEX(r, pop), IDEX(r, pop), IDEX(r, pop),
```

sub:

```
  /* 0x28 */    IDEXW(G2E, sub, 1), IDEX(G2E, sub), IDEXW(E2G, sub, 1), IDEX(E2G, sub),

  /* 0x2c */    IDEXW(I2a, sub, 1), IDEX(I2a, sub), EMPTY, EMPTY,
```

xor:

```
  /* 0x30 */    IDEXW(G2E, xor, 1), IDEX(G2E, xor), IDEXW(E2G, xor, 1), IDEX(E2G, xor),
  /* 0x34 */    EMPTY, IDEX(I2a, xor), EMPTY, EMPTY,
```

ret:

```
   /* 0xc0 */    EMPTY, EMPTY, EMPTY, EX(ret),
```

```
root@zhaoweikang:/home/zhaoweikang/ics201//nexus-am/tests/cputest# make ARCH=x86
-nemu ALL=dummy run
Building dummy [x86-nemu]
Building am [x86-nemu]
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/monitor.c,65,load_img] The image is /home/zhaoweikang/ics2017/nexus
-am/tests/cputest/build/dummy-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 22:30:21, Apr 18 2018
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

最后出现了 **HIT GOOD TRAP** …说明成功实现了相关指令

git log 记录

```
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo git status
位于分支 master
尚未暂存以备提交的变更：
  （使用 "git add <文件>..." 更新要提交的内容）
  （使用 "git checkout -- <文件>..." 丢弃工作区的改动）

        修改：     include/cpu/reg.h
        修改：     include/cpu/rtl.h
        修改：     src/cpu/decode/decode.c
        修改：     src/cpu/exec/all-instr.h
        修改：     src/cpu/exec/arith.c
        修改：     src/cpu/exec/control.c
        修改：     src/cpu/exec/data-mov.c
        修改：     src/cpu/exec/exec.c
        修改：     src/cpu/exec/logic.c
        修改：     src/monitor/monitor.c

修改尚未加入提交（使用 "git add" 和/或 "git commit -a"）
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo git add .
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo git commit --allow-empty
[master 7846ce6] fix bug for pa2.1
 10 files changed, 269 insertions(+), 55 deletions(-)
zhaoweikang@zhaoweikang:~/ics2017/nemu$ sudo git log
commit 7846ce66889b15109707f214bbbeec3551a79c72
Author: 161630220-Zhao Weikang <2875206963@qq.com>
Date:   Thu Apr 19 21:03:50 2018 +0800

    fix bug for pa2.1

commit b4b6ebd8315401ddbed4c8fde498019184926ca2
Author: tracer-ics2017 <tracer@njuics.org>
Date:   Thu Apr 19 20:51:52 2018 +0800

    >  run
    161630220
    root
    Linux zhaoweikang 4.9.0-6-686-pae #1 SMP Debian 4.9.82-1+deb9u3 (2018-03-02)
 i686 GNU/Linux
     20:51:52 up 54 min,  1 user,  load average: 0.40, 0.20, 0.13
    b61eb6bcc15996b27c500a35736016cc1cda3bc

commit 75b583c0daeb1cc7322650d06c3d64c2b80303ff
Author: tracer-ics2017 <tracer@njuics.org>
Date:   Thu Apr 19 20:51:52 2018 +0800

    >  compile
:
```