南京航空航天大学《计算机组成原理Ⅱ课程设计》报告

- 姓名：邵震哲
- 班级：1620204
- 学号：162020130
- 报告阶段：PA2.2&2.3
- 完成日期：2022.5.14
- 本次实验，操作题中的捕捉死循环（加分项）未完成，其他均完成

思考题

1. 什么是 API

   API（Application Programming Interface,应用程序编程接口）是一些预先定义的函数，目的是提供应用程序与开发人员基于某软件或硬件得以访问一组例程的能力，而又无需访问源码，或理解内部工作机制的细节

2. AM 属于硬件还是软件?

   AM属于软件。我认为AM 和操作系统一样。AM是一个抽象计算机模型，通过一组API实现对计算机底层细节的抽象，为程序运行提供最基本的软件支持，是最贴近硬件的软件。操作系统是位于硬件与软件之间，而AM也是位于NEMU（硬件）与软件之间，他们的功能都是通过API实现对硬件的抽象。

3. 堆和栈在哪里?

   如果放到可执行文件里，读取、操作数据的速度会很慢。堆和栈需要在内存中动态申请。

4. 回忆运行过程

   1.根据`ARCH=x86-nemu`让程序编译到到`x86-nemu`中

   2.再根据`ALL=dummy`可知`make run`指令调用了`nexus-am/am/arch/x86-nemu/img/run`启动`nemu`来运行`dummy.c`

5. 神奇的eflags（2）

```
+-----+-----+-------------------------------------+
|  SF |  OF |                 实例                |
+-----+-----+-------------------------------------+
|  0  |  0  |                2 - 1                |
+-----+-----+-------------------------------------+
|  0  |  1  |          0xf0000000-0x00000001      |
+-----+-----+-------------------------------------+
|  1  |  0  |          0x80000001-0x00000001      |
+-----+-----+-------------------------------------+
|  1  |  1  |          0x0f000000-0x00000001      |
+-----+-----+-------------------------------------+
```

6. 这是巧合吗?

- `ja`

  无符号整数`op1>op2`，`CF=0`且`ZF=0`

- `jb`

  无符号整数`op1<op2`，`CF=1`且`ZF=0`

- `jg`

  带符号整数`op1>op2`，`SF=OF`且`ZF=0`

- `jl`

  带符号整数`op1<op2`，`SF≠OF`

7. nemu的本质

```
lable1:
    x = x - 1
    a = a + 1
    jne x， lable1

lable2:
    y = y - 1
    a = a + 1
    jne y， lable2
```

nemu还需要输入输出功能和图形界面来进行交互

8. 设备是如何工作的?

   我会设置一个I/O接口，用于连接CPU与外部设备。CPU通过接口发送指令给设备，设备接受指令并执行后把结果通过接口传回CPU。这样CPU与外部设备不需要知道对方是怎么工作的，只需要和接口通信即可。

9. CPU 需要知道设备是如何工作的吗?

   不需要。接口相当于API，CPU通过接口发送指令给设备，然后等待设备传回结果从接口接收即可，不需要知道设备如何运作。设备对于CPU相当于一个黑盒，只需要知道怎么用即可，不关心内部怎么工作。

10. 什么是驱动?

    驱动，是指驱动计算机里软件的程序。驱动程序全称设备驱动程序，是添加到操作系统中的特殊程序，其中包含有关硬件设备的信息。此信息能够使计算机与相应的设备进行通信。驱动程序是硬件厂商根据操作系统编写的配置文件，可以说没有驱动程序，计算机中的硬件就无法工作。

    驱动是操作系统与硬件之间的桥梁，操作系统有了驱动，才能调度硬件。一般应用程序在操作系统之上，利用操作系统提供的API完成相应的任务。

11. cpu知道吗?

    不需要。只需要把指定地址上的值定为指定的值即可。

12. 再次理解volatile

    O2优化下编译

    ```
    gcc -O2 -o fun fun.c
    ```

    查看反汇编

    ```
    objdump -s -d fun > fun.txt
    ```

    添加`volatile`关键字的反汇编

```
shaozhenzhe@Debian: ~/helloproject
    118e:       66 90                   xchg   %ax,%ax

00001190 <frame_dummy>:
    1190:       e9 5b ff ff ff          jmp    10f0 <register_tm_clones>

00001195 <__x86.get_pc_thunk.dx>:
    1195:       8b 14 24                mov    (%esp),%edx
    1198:       c3                      ret
    1199:       66 90                   xchg   %ax,%ax
    119b:       66 90                   xchg   %ax,%ax
    119d:       66 90                   xchg   %ax,%ax
    119f:       90                      nop

000011a0 <fun>:
    11a0:       c6 05 00 80 04 08 00    movb   $0x0,0x8048000
    11a7:       8d b4 26 00 00 00 00    lea    0x0(%esi,%eiz,1),%esi
    11ae:       66 90                   xchg   %ax,%ax
    11b0:       0f b6 05 00 80 04 08    movzbl 0x8048000,%eax
    11b7:       3c ff                   cmp    $0xff,%al
    11b9:       75 f5                   jne    11b0 <fun+0x10>
    11bb:       c6 05 00 80 04 08 33    movb   $0x33,0x8048000
    11c2:       c6 05 00 80 04 08 34    movb   $0x34,0x8048000
    11c9:       c6 05 00 80 04 08 36    movb   $0x36,0x8048000
    11d0:       c3                      ret
    11d1:       66 90                   xchg   %ax,%ax
    11d3:       66 90                   xchg   %ax,%ax
    11d5:       66 90                   xchg   %ax,%ax
    11d7:       66 90                   xchg   %ax,%ax
    11d9:       66 90                   xchg   %ax,%ax
    11db:       66 90                   xchg   %ax,%ax
    11dd:       66 90                   xchg   %ax,%ax
    11df:       90                      nop

000011e0 <__libc_csu_init>:
    11e0:       55                      push   %ebp
    11e1:       57                      push   %edi
    11e2:       56                      push   %esi
                                                                    345,1          87%
```

不添加`volatile`关键字的反汇编

```
shaozhenzhe@Debian: ~/helloproject
    1185:       c9                      leave
    1186:       c3                      ret
    1187:       8d b4 26 00 00 00 00    lea    0x0(%esi,%eiz,1),%esi
    118e:       66 90                   xchg   %ax,%ax

00001190 <frame_dummy>:
    1190:       e9 5b ff ff ff          jmp    10f0 <register_tm_clones>

00001195 <__x86.get_pc_thunk.dx>:
    1195:       8b 14 24                mov    (%esp),%edx
    1198:       c3                      ret
    1199:       66 90                   xchg   %ax,%ax
    119b:       66 90                   xchg   %ax,%ax
    119d:       66 90                   xchg   %ax,%ax
    119f:       90                      nop

000011a0 <fun>:
    11a0:       c6 05 00 80 04 08 00    movb   $0x0,0x8048000
    11a7:       eb fe                   jmp    11a7 <fun+0x7>
    11a9:       66 90                   xchg   %ax,%ax
    11ab:       66 90                   xchg   %ax,%ax
    11ad:       66 90                   xchg   %ax,%ax
    11af:       90                      nop

000011b0 <__libc_csu_init>:
    11b0:       55                      push   %ebp
    11b1:       57                      push   %edi
    11b2:       56                      push   %esi
    11b3:       53                      push   %ebx
    11b4:       e8 e7 fe ff ff          call   10a0 <__x86.get_pc_thunk.bx>
    11b9:       81 c3 47 2e 00 00       add    $0x2e47,%ebx
    11bf:       83 ec 0c                sub    $0xc,%esp
    11c2:       8b 6c 24 28             mov    0x28(%esp),%ebp
    11c6:       e8 35 fe ff ff          call   1000 <_init>
    11cb:       8d b3 f8 fe ff ff       lea    -0x108(%ebx),%esi
    11d1:       8d 83 f4 fe ff ff       lea    -0x10c(%ebx),%eax
    11d7:       29 c6                   sub    %eax,%esi
                                                                    339,1          89%
```

若不加`volatile`，第`11a7`行会陷入死循环。

13. hello world运行在哪里？

    不一样。这里在AM层，程序设计课编程上的运行在内存等硬件层上

14. 如何检测很多个键同时被按下？

    当按下一个键的时候， 键盘将会发送该键的通码; 当释放一个键的时候， 键盘将会发送该键的断码。每当用户敲下/释放按键时， 将会把相应的键盘码放入数据寄存器， 同时把状态寄存器的标志设置为 `1`， 表示有按键事件发生. CPU 可以通过端口 I/O 访问这些寄存器， 获得键盘码。每个按键的通码断码都不同，因此计算机可以识别出同时按下的不同的按键。

15. 编译与链接 I

    去掉`static`，没有报错

```
shaozhenzhe@Debian:~/ics2022.bak/nemu$ make run
+ CC src/cpu/exec/control.c
+ CC src/cpu/exec/exec.c
+ CC src/cpu/exec/special.c
+ CC src/cpu/exec/data-mov.c
+ CC src/cpu/exec/logic.c
+ CC src/cpu/exec/prefix.c
+ CC src/cpu/exec/system.c
+ CC src/cpu/exec/cc.c
+ CC src/cpu/exec/arith.c
+ CC src/cpu/intr.c
+ CC src/cpu/decode/decode.c
+ CC src/cpu/decode/modrm.c
+ LD build/nemu
./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: 09:52:08, May 13 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x00100026

(nemu)
```

    去掉`inline`，报错`defined but not used`

```
shaozhenzhe@Debian:~/ics2022.bak/nemu$ make run
+ CC src/cpu/exec/control.c
In file included from ./include/cpu/decode.h:6,
                 from ./include/cpu/exec.h:9,
                 from src/cpu/exec/control.c:1:
./include/cpu/rtl.h:46:13: error: 'rtl_mul' defined but not used [-Werror=unused-funct
ion]
 static void rtl_mul(rtlreg_t* dest_hi, rtlreg_t* dest_lo, const rtlreg_t* src1, const
 rtlreg_t* src2) {
             ^~~~~~~
cc1: all warnings being treated as errors
make: *** [Makefile:25: build/obj/cpu/exec/control.o] Error 1
shaozhenzhe@Debian:~/ics2022.bak/nemu$
```

    该函数仅用`static`修饰，则为静态函数，只能被该函数所在文件引用，然而该文件并没用其他函数使用该函数，因此导致`defined but not used`

    去掉`static`和`inline`

```
shaozhenzhe@Debian:~/ics2022.bak/nemu$ make run
+ CC src/cpu/exec/control.c
+ CC src/cpu/exec/exec.c
+ CC src/cpu/exec/special.c
+ CC src/cpu/exec/data-mov.c
+ CC src/cpu/exec/logic.c
+ CC src/cpu/exec/prefix.c
+ CC src/cpu/exec/system.c
+ CC src/cpu/exec/cc.c
+ CC src/cpu/exec/arith.c
+ CC src/cpu/intr.c
+ CC src/cpu/decode/decode.c
+ CC src/cpu/decode/modrm.c
+ LD build/nemu
/usr/bin/ld: build/obj/cpu/exec/exec.o: in function `rtl_mul':
/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/rtl.h:47: multiple definition of `rtl
_mul'; build/obj/cpu/exec/control.o:/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/r
tl.h:47: first defined here
/usr/bin/ld: build/obj/cpu/exec/special.o: in function `rtl_mul':
/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/rtl.h:47: multiple definition of `rtl
_mul'; build/obj/cpu/exec/control.o:/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/r
tl.h:47: first defined here
/usr/bin/ld: build/obj/cpu/exec/data-mov.o: in function `rtl_mul':
/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/rtl.h:47: multiple definition of `rtl
_mul'; build/obj/cpu/exec/control.o:/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/r
tl.h:47: first defined here
/usr/bin/ld: build/obj/cpu/exec/logic.o: in function `rtl_mul':
/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/rtl.h:47: multiple definition of `rtl
_mul'; build/obj/cpu/exec/control.o:/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/r
tl.h:47: first defined here
/usr/bin/ld: build/obj/cpu/exec/prefix.o: in function `rtl_mul':
/home/shaozhenzhe/ics2022.bak/nemu/./include/cpu/rtl.h:47: multiple definition of `rtl
```

当多个文件包含同一个头文件时，而头文件中没有加上条件编译，就会独立的解释，然后每个文件生成独立的标示符。在编译器连接时，就会将工程中所有的符号整合在一起，由于文件中有重名变量，于是就出现了重复定义的错误。

16. 编译与链接Ⅱ（10分）

    1. 29个

shaozhenzhe@Debian: ~/ics2022.bak/nemu

```
make: *** [Makefile:25: build/obj/memory/memory.o] Error 1
shaozhenzhe@Debian:~/ics2022.bak/nemu$ make run
+ CC src/memory/memory.c
+ CC src/device/timer.c
+ CC src/device/io/port-io.c
+ CC src/device/io/mmio.c
+ CC src/device/device.c
+ CC src/device/vga.c
+ CC src/device/serial.c
+ CC src/device/keyboard.c
+ CC src/cpu/exec/control.c
+ CC src/cpu/exec/exec.c
+ CC src/cpu/exec/special.c
+ CC src/cpu/exec/data-mov.c
+ CC src/cpu/exec/logic.c
+ CC src/cpu/exec/prefix.c
+ CC src/cpu/exec/system.c
+ CC src/cpu/exec/cc.c
+ CC src/cpu/exec/arith.c
+ CC src/cpu/reg.c
+ CC src/cpu/intr.c
+ CC src/cpu/decode/decode.c
+ CC src/cpu/decode/modrm.c
+ CC src/monitor/debug/ui.c
+ CC src/monitor/debug/watchpoint.c
+ CC src/monitor/debug/expr.c
+ CC src/monitor/monitor.c
+ CC src/monitor/diff-test/gdb-host.c
+ CC src/monitor/diff-test/diff-test.c
+ CC src/monitor/diff-test/protocol.c
+ CC src/monitor/cpu-exec.c
+ LD build/nemu
./build/nemu -l ./build/nemu-log.txt
[src/monitor/monitor.c,47,load_default_img] No image is given. Use the default build-in image.
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 11:15:22, May 14 2022
For help, type "help"
(nemu)
```

    有29个文件重新编译，因此有29个变量实体。

    2. 58

    两次定义的符号没有赋初值，因此都是弱符号，编译器允许弱符号多次定义且编译器会把这两次当作两个不同的符号对待。因此又多了29个变量实体，一共58个。

3. 初始化

```
shaozhenzhe@Debian:~/ics2022.bak/nemu$ make run
+ CC src/memory/memory.c
In file included from ./include/common.h:10,
                 from ./include/nemu.h:4,
                 from src/memory/memory.c:1:
./include/debug.h:6:21: error: redefinition of 'dummy'
 volatile static int dummy=0;
                     ^~~~~
In file included from ./include/nemu.h:4,
                 from src/memory/memory.c:1:
./include/common.h:3:21: note: previous definition of 'dummy' was here
 volatile static int dummy=0;
                     ^~~~~
make: *** [Makefile:25: build/obj/memory/memory.o] Error 1
shaozhenzhe@Debian:~/ics2022.bak/nemu$
```

变量赋初值后则成为强符号，编译器不允许强符号多次定义，因此报错。

17. I/O 端口与接口（10分）

1. 地址范围：

   1K=2^10，端口范围 `0000H` ~ `0400H`

   16根地址总线，寻址范围也就是2^16，地址范围 `0000H` ~ `FFFFH`

2. I/O三种控制方式：程序直接控制、终端控制、DMA控制。

   首先DMA控制器初始化，然后发送“启动DMA传送”命令以启动外设进行I/O操作，发送完“启动DMA传送”命令后， CPU转去执行其他进程，而请求I/O的用户进程被阻塞。在CPU执行其他进程的过程中，DMA控制器外设和主存进行数据交换。DMA控制器每完成一个数据的传送，就将字计数器减1，并修改主存地址，当字计数器为0时，完成所有I/O操作，此时，DMA控制器将向CPU发送“DMA完成”中断请求，CPU检测到后调出相应的中断服务程序执行。CPU在中断服务程序中，解除用户进程的阻塞状态而使用户进程进入就绪序列，然后中断返回，再回到被打断的进程继续执行。（来自课本）

   常见于硬盘。

18. git log截图

shaozhenzhe@Debian: ~/ics2022/nemu

```
33f39ea (HEAD -> pa2) >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19
.208-1 (2021-09-29) i686 GNU/Linux  11:24:10 up 35 min,  2 users,  load average: 0.04, 0.02, 0.0
2 afd09a2fbc74493fc8aca43dc32e70c53770649
55a49d5 >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  11:05:55 up 17 min,  2 users,  load average: 0.01, 0.12, 0.08 856989f169d4
79625fd748df4ce7eebb2376ac58
62ff1b9 >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  11:01:54 up 13 min,  2 users,  load average: 0.07, 0.18, 0.09 27337fe68dea
f402be4b17c730719c651f0e2154
16c729f >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  10:58:16 up 9 min,  2 users,  load average: 0.50, 0.14, 0.05 9b5fd58b78696
643e288e967366f92063ca376da
3534a1f >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  10:57:46 up 9 min,  2 users,  load average: 0.00, 0.03, 0.01 e6b9c7cbaca9d
4e4923f47012812be678c496c4e
8294a74 >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  10:50:24 up 1 min,  2 users,  load average: 0.10, 0.04, 0.01 c0f99bf34fbbd
b654fc50edfe5f46322f5e0976
4abcf7e >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  22:21:21 up  4:57,  2 users,  load average: 0.00, 0.00, 0.00 5cf3098e2339e
ea52dd93d2739710f6020f45b5b
7ffee6b PA2.2&2.3 finished
af0594e finished task 8
62142df >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  17:44:35 up 20 min,  2 users,  load average: 0.50, 0.41, 0.20 7fd92a414499
46771b404c46ec5abee32d2208ef
e8630f9 >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  17:42:24 up 18 min,  2 users,  load average: 0.17, 0.17, 0.10 729edc3c9d60
b4352512eafdfd825cc390911a
d8ceab5 finished task 7
563488c >  run 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (2021-0
9-29) i686 GNU/Linux  17:40:04 up 16 min,  2 users,  load average: 0.00, 0.04, 0.05 7b6383ee8d4c
88ae6f6e36764c7f876f61167378
80e6273 >  compile 162020130 shaozhenzhe Linux Debian 4.19.0-18-686 #1 SMP Debian 4.19.208-1 (20
21-09-29) i686 GNU/Linux  17:40:04 up 16 min,  2 users,  load average: 0.00, 0.04, 0.05 b4432168
81ee4926a0aac86d3d1297707bd23885
cde0710 finished task 6
:
```

实验内容

实现剩余所有 x86 指令（40 分）

- 约有 40 个（有的只需要填表），以跑通所有测试用例为准；
- 指令实现的顺序为：
  - 运行一个新的测试用例；
  - 该测试用例新出现了某些指令；
  - 逐个实现每个出现的指令；
  - 本测试样例成功运行，开始下一个测试用例直至全部用例通过

add.c

8d(lea)

查手册填表即可

```
  /* 0x8c */    EMPTY, IDEX(lea_M2G, lea), EMPTY, EMPTY,
```

这一条指令忘记截图

83 e4(and)

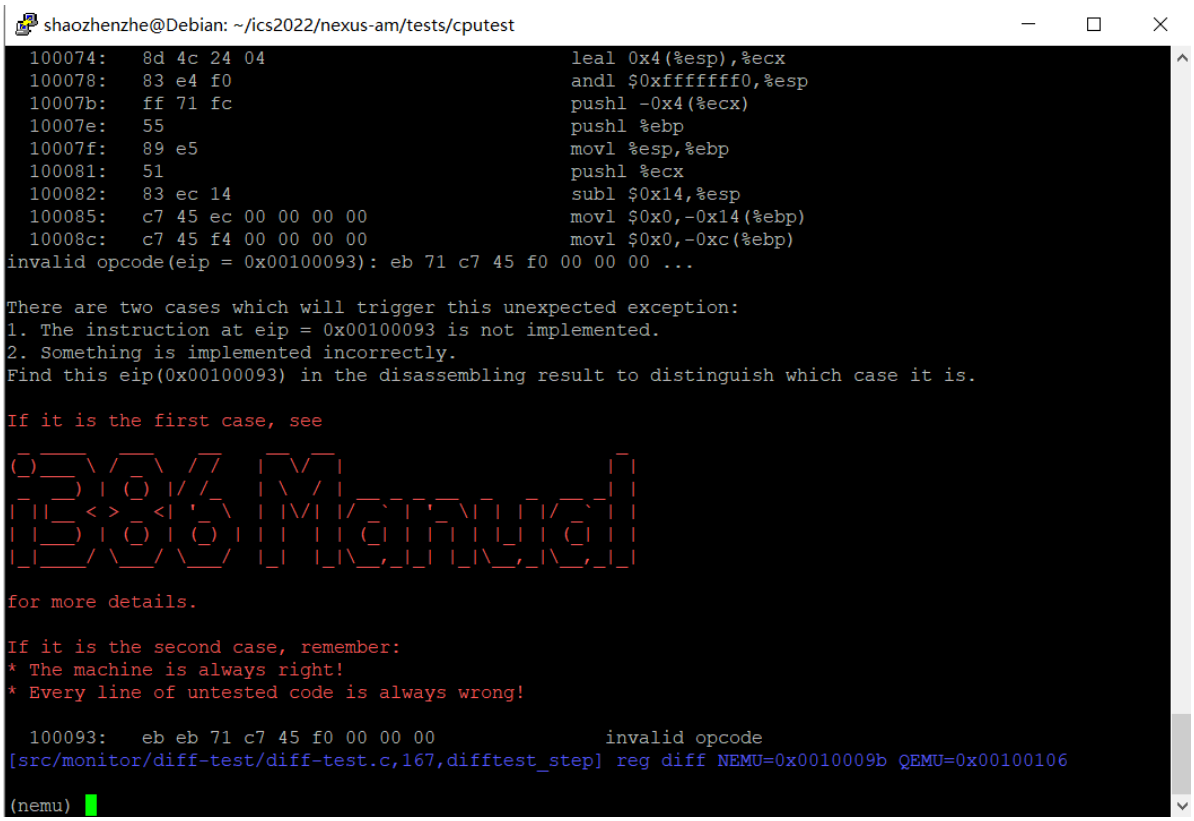
查表得知译码函数为SI2E，需要完善SI实现位扩展

```
static inline make_DopHelper(SI) {
  assert(op->width == 1 || op->width == 4);

  op->type = OP_TYPE_IMM;

  /* TODO: Use instr_fetch() to read `op->width' bytes of memory
   * pointed by `eip'. Interpret the result as a signed immediate,
   * and assign it to op->simm.
   *
   op->simm = ???
   */
  //TODO();
  op->simm = instr_fetch(eip, op->width);

  rtl_li(&op->val, op->simm);

  rtl_sext(&op->val, &op->val, op->width);
  op->simm = op->val;


#ifdef DEBUG
  snprintf(op->str, OP_STR_SIZE, "$0x%x", op->simm);
#endif
}
```

执行函数是`and`，调用rtl_and相与并且更新标志位即可

```
make_EHelper(and) {
  //TODO();
  rtl_and(&id_dest->val, &id_dest->val, &id_src->val);
  operand_write(id_dest, &id_dest->val);

  rtl_set_CF(&tzero);//CF=0
  rtl_set_OF(&tzero);//OF=0
  rtl_update_ZFSF(&id_dest->val, id_dest->width);//更新ZF、SF

  print_asm_template2(and);
}
```

运行结果

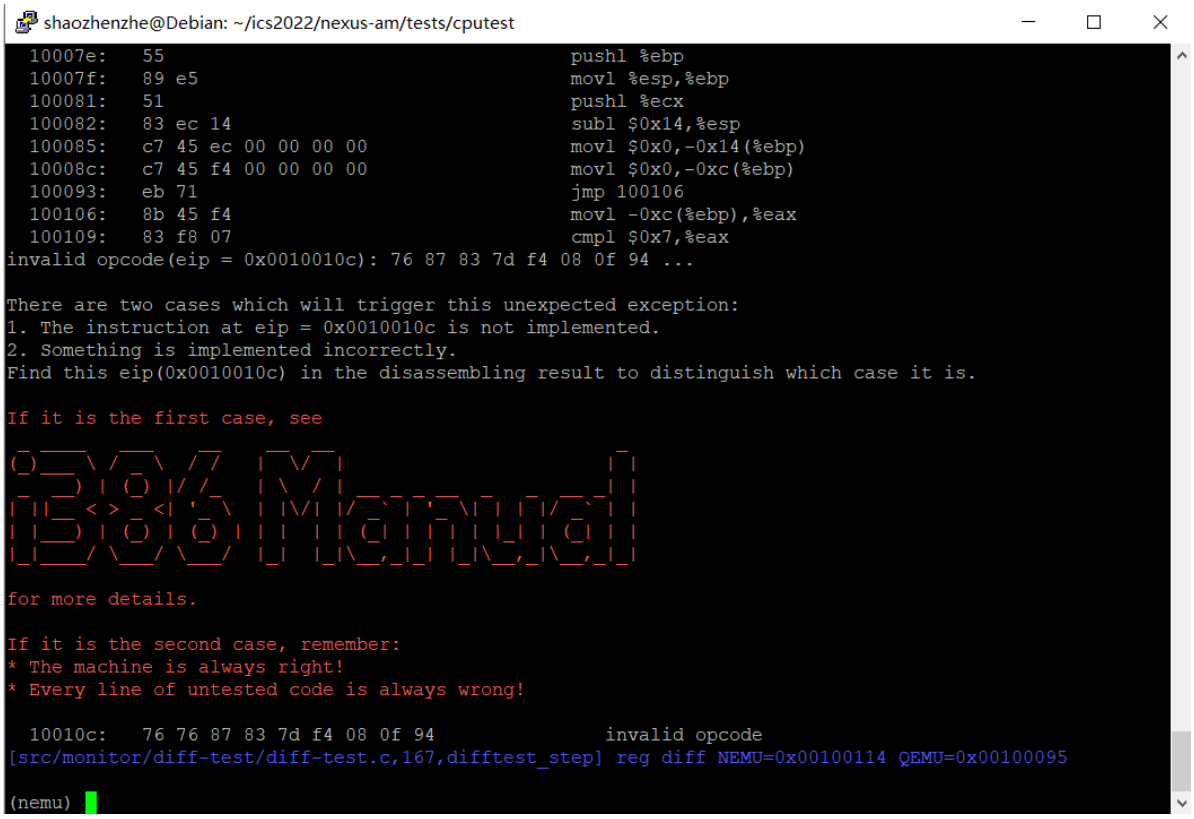


eb(jmp)

译码函数`J`，执行函数`jmp`

```
  /* 0xe8 */    IDEX(I, call), EMPTY, EMPTY, IDEXW(J, jmp, 1),
```

运行结果



76(jbe)

译码函数`J`，执行函数`jcc`

```
/* 0x74 */    EMPTY, EMPTY, IDEXW(J, jcc, 1), EMPTY,
```

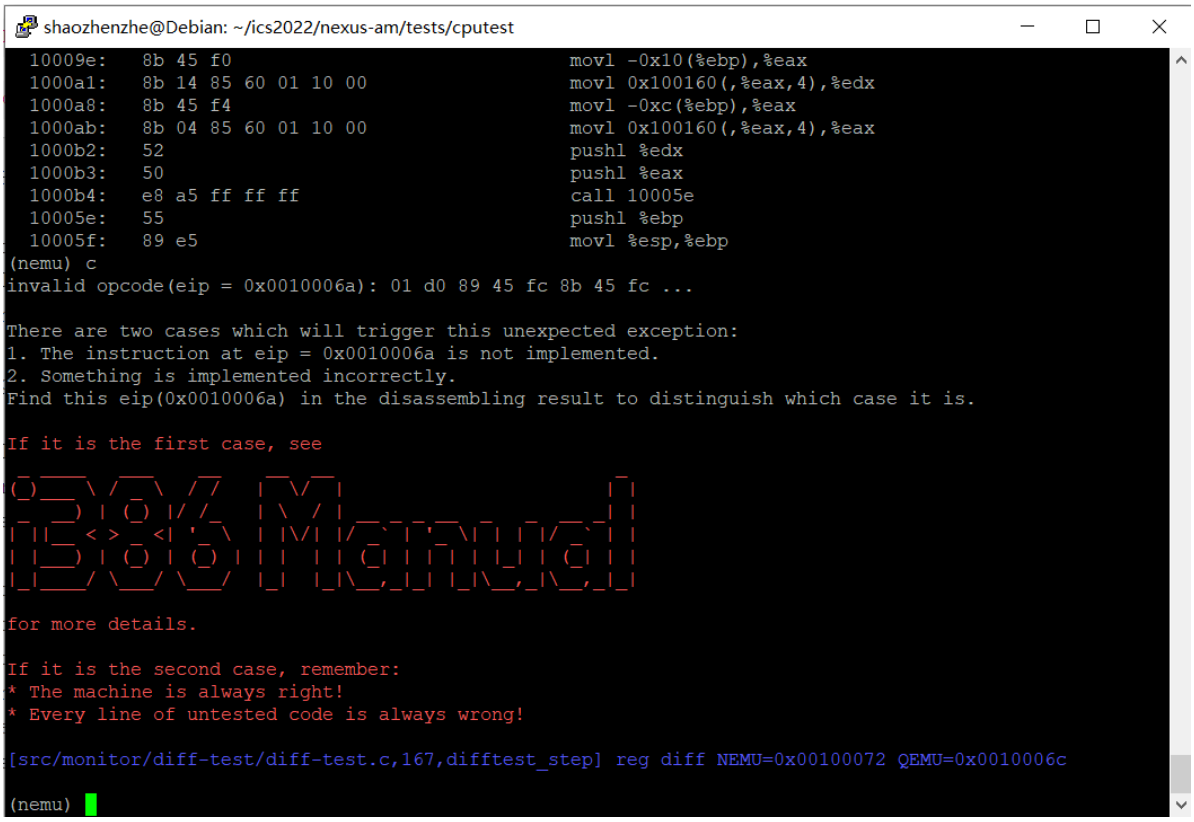
`make_EHelper(jcc)`需要实现`rtl_setcc()`，根据手册实现`eflags`的读取即可

`nemu/src/cpu/exec/cc.c`

```
void rtl_setcc(rtlreg_t* dest, uint8_t subcode) {
  bool invert = subcode & 0x1;
  enum {
    CC_O, CC_NO, CC_B,  CC_NB,
    CC_E, CC_NE, CC_BE, CC_NBE,
    CC_S, CC_NS, CC_P,  CC_NP,
    CC_L, CC_NL, CC_LE, CC_NLE
  };

  // TODO: Query EFLAGS to determine whether the condition code is satisfied.
  // dest <- ( cc is satisfied ? 1 : 0)
  switch (subcode & 0xe) {
    case CC_O:
        rtl_get_OF(dest);
        break;
    case CC_B:
        rtl_get_CF(dest);
        break;
    case CC_E:
        rtl_get_ZF(dest);
        break;
    case CC_BE:
        rtl_get_CF(&t0);
        rtl_get_ZF(&t1);
```

```
            rtl_or(dest, &t0, &t1);
            break;
        case CC_S:
            rtl_get_SF(dest);
            break;
        case CC_L:
            rtl_get_SF(&t0);
            rtl_get_OF(&t1);
            rtl_xor(dest, &t0, &t1);
            break;
        case CC_LE:
            rtl_get_ZF(&t0);
            rtl_get_SF(&t1);
            rtl_get_OF(&t2);
            rtl_xor(&t3, &t1, &t2);
            rtl_or(dest, &t0, &t3); //带符号数的小于等于，ZF=1或者SF不等于OF
            break;
          //TODO();
        default: panic("should not reach here");
        case CC_P: panic("n86 does not have PF");
      }

      if (invert) {
        rtl_xori(dest, dest, 0x1);
      }
    }
```

运行结果



01(add)

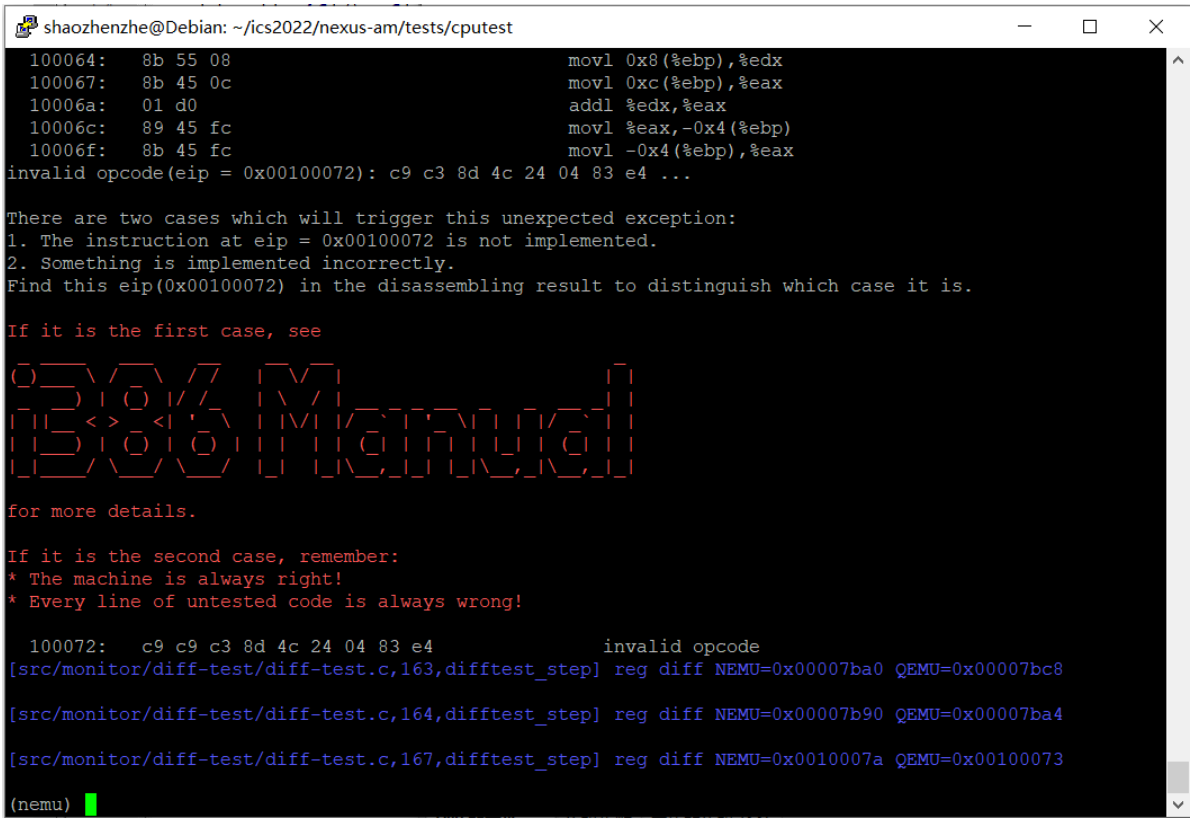
译码函数`G2E`，执行函数`add`，填表时顺便将0x00-0x05全部填好

```
  /* 0x00 */    IDEXW(G2E, add, 1), IDEX(G2E, add), IDEXW(E2G, add, 1),
IDEX(E2G, add),
  /* 0x04 */    IDEXW(I2a, add, 1), IDEX(I2a, add), EMPTY, EMPTY,
```

完善执行函数`add`，模仿`adc`，去掉CF即可

```
make_EHelper(add) {
  //TODO();

  rtl_add(&t2, &id_dest->val, &id_src->val);
  rtl_sltu(&t3, &t2, &id_dest->val);
  //rtl_get_CF(&t1);
  //rtl_add(&t2, &t2, &t1);
  operand_write(id_dest, &t2);

  rtl_update_ZFSF(&t2, id_dest->width);

  rtl_sltu(&t0, &t2, &id_dest->val);
  rtl_or(&t0, &t3, &t0);
  rtl_set_CF(&t0);

  rtl_xor(&t0, &id_dest->val, &id_src->val);
  rtl_not(&t0);
  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(add);
}
```

运行结果



c9(leave)

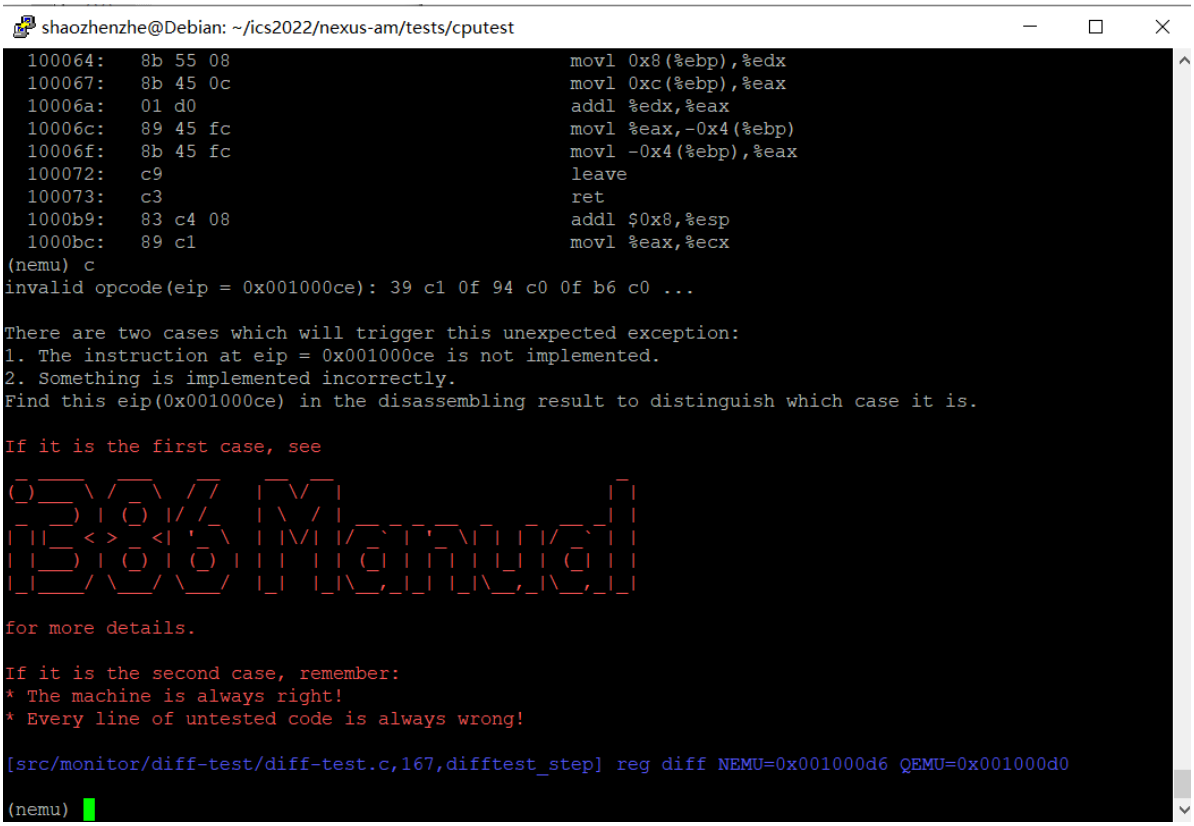
无译码函数，执行函数`leave`

```
  /* 0xc8 */    EMPTY, EX(leave), EMPTY, EMPTY,
```

完善执行函数，先把ebp的值赋给esp，再弹栈

```
make_EHelper(leave) {
  //TODO();

  rtl_mv(&cpu.esp, &cpu.ebp);
  rtl_pop(&cpu.ebp);
  print_asm("leave");
}
```

运行结果



39(cmp)

根据手册直接填表

```
  /* 0x38 */    IDEXW(G2E, cmp, 1), IDEX(G2E, cmp), IDEXW(E2G, cmp, 1),
IDEX(E2G, cmp),
  /* 0x3c */    IDEXW(I2a, cmp, 1), IDEX(I2a, cmp), EMPTY, EMPTY,
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
  10006a:   01 d0                                   addl %edx,%eax
  10006c:   89 45 fc                                movl %eax,-0x4(%ebp)
  10006f:   8b 45 fc                                movl -0x4(%ebp),%eax
  100072:   c9                                      leave
  100073:   c3                                      ret
  1000b9:   83 c4 08                                addl $0x8,%esp
  1000bc:   89 c1                                   movl %eax,%ecx
(nemu) c
invalid opcode(eip = 0x001000d0): 0f 94 c0 0f b6 c0 83 ec ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x001000d0 is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x001000d0) 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!

[src/monitor/diff-test/diff-test.c,159,difftest_step] reg diff NEMU=0x00000000 QEMU=0x00000001

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x001000d8 QEMU=0x001000d3

(nemu)
```

0f 94(set)

0f指向第二个表，查手册得到set，译码函数`E`，执行函数`setcc`

```
    /* 0x94 */     IDEXW(E, setcc, 1), EMPTY, EMPTY, EMPTY,
```

运行结果

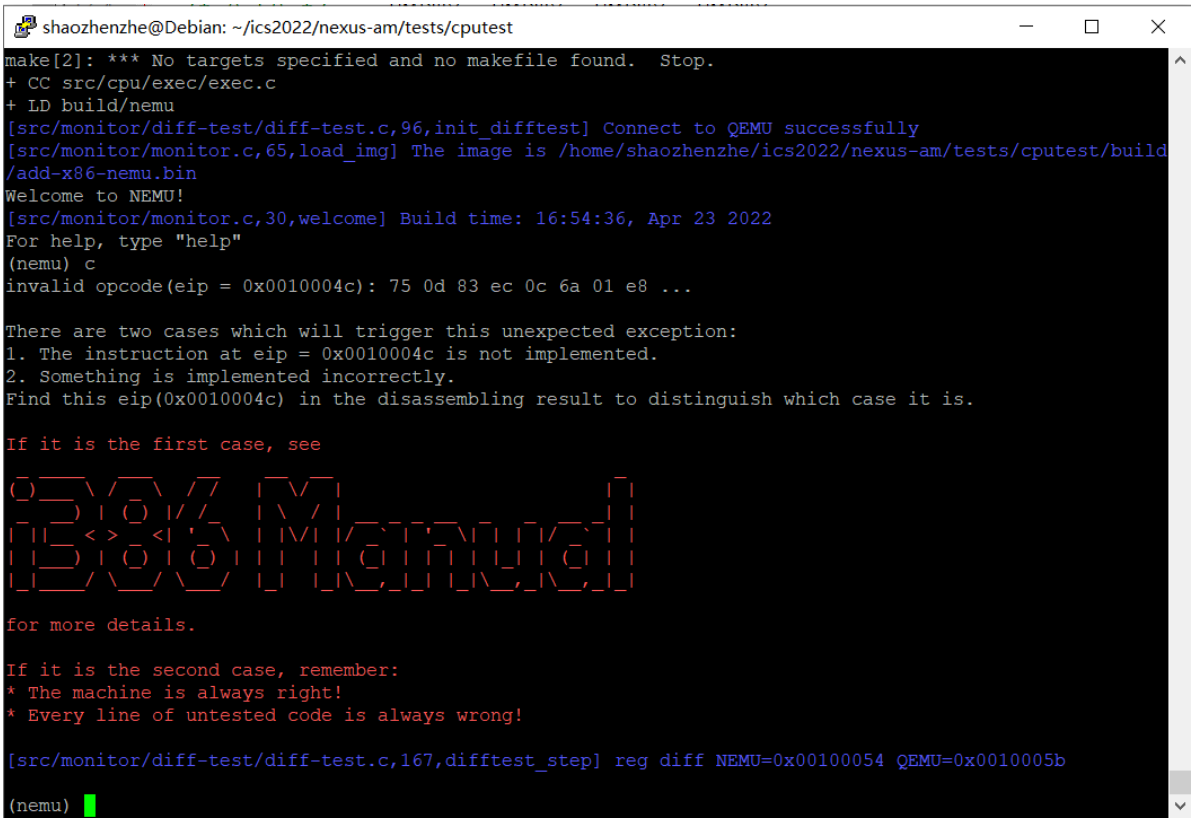
```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cputest/build
/add-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x001000d3): 0f b6 c0 83 ec 0c 50 e8 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x001000d3 is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x001000d3) 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!

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x001000db QEMU=0x001000d6

(nemu)
```

0f b6(movzbl)

看第二个表，译码函数`mov_E2G`，执行函数`movxz`，顺便把b7也填了

```
/* 0xb4 */ EMPTY, EMPTY, IDEXW(mov_E2G, movzx, 1), IDEXW(mov_E2G, movzx, 2),
```

运行结果

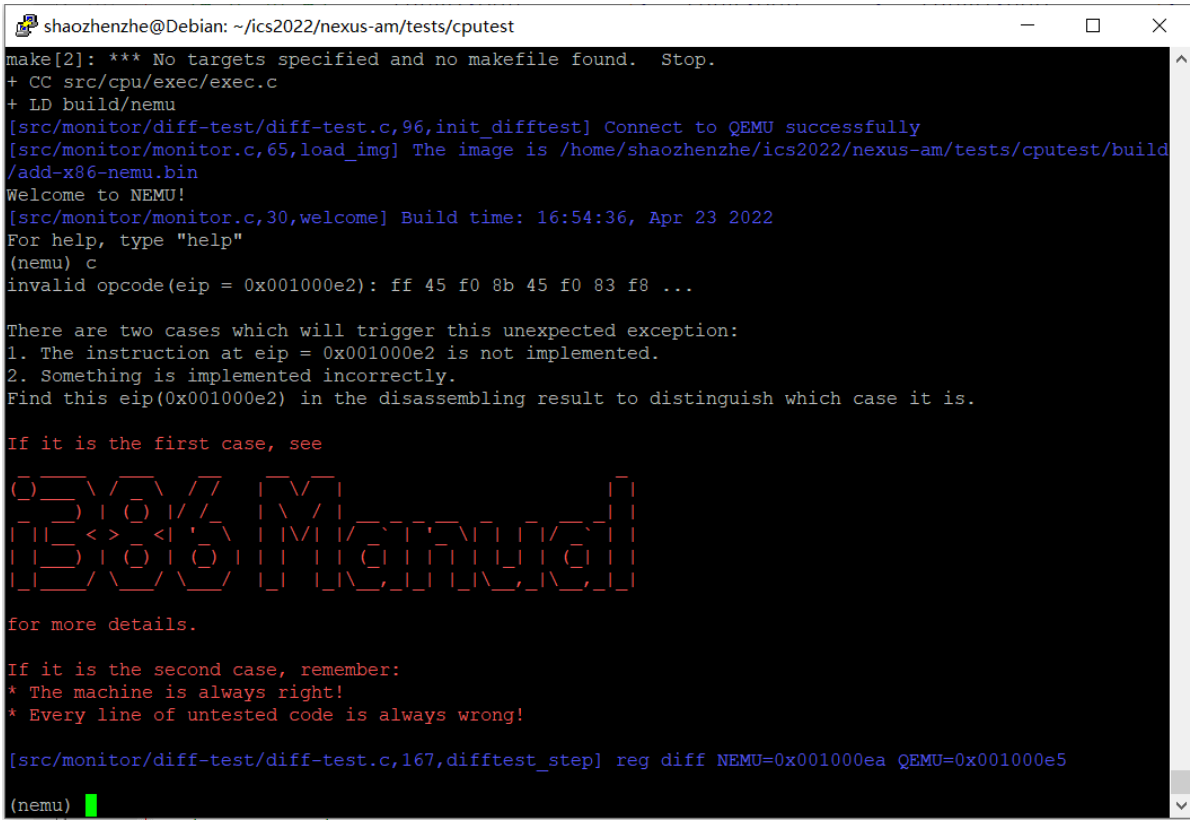


75(jnz)

和 `jbe` 类似

```
/* 0x74 */    EMPTY, IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), EMPTY,
```

运行结果



ff 45(inc)

ff是gp5，根据45得知是`inc`

```
make_group(gp5,
    EX(inc), EMPTY, EMPTY, EMPTY,
    EMPTY, EMPTY, EX(push), EMPTY)
```

模仿`add`，改为加一即可

```
make_EHelper(inc) {
  //TODO();
  rtlreg_t temp = 1;
  rtl_add(&t2, &id_dest->val, &temp);
  rtl_sltu(&t3, &t2, &id_dest->val);
  //rtl_get_CF(&t1);
  //rtl_add(&t2, &t2, &t1);
  operand_write(id_dest, &t2);

  rtl_update_ZFSF(&t2, id_dest->width);

  rtl_sltu(&t0, &t2, &id_dest->val);
  rtl_or(&t0, &t3, &t0);
  rtl_set_CF(&t0);

  rtl_xor(&t0, &id_dest->val, &id_src->val);
  rtl_not(&t0);
  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_template1(inc);
}
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=add run
Building add [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cputest/build
/add-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

add-longlong

e9(jmp)

```
/* 0xe8 */    IDEX(I, call), IDEX(J, jmp), EMPTY, IDEXW(J, jmp, 1),
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cputest/build
/add-longlong-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x00100158): 0f 86 68 ff ff ff b8 00 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x00100158 is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x00100158) 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!

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x00100160 QEMU=0x001000c6

(nemu)
```

0f 86(jcc)

看第二个表，和之前类似，译码函数`J`，执行函数`jcc`，这里把0x80-0x8f都填完了，都是`jcc`

```
/* 0x80 */    IDEX(J, jcc), IDEX(J, jcc), IDEX(J, jcc), IDEX(J, jcc),
/* 0x84 */    IDEX(J, jcc), IDEX(J, jcc), IDEX(J, jcc), IDEX(J, jcc),
/* 0x88 */    IDEX(J, jcc), IDEX(J, jcc), IDEX(J, jcc), IDEX(J, jcc),
/* 0x8c */    IDEX(J, jcc), IDEX(J, jcc), IDEX(J, jcc), IDEX(J, jcc),
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cputest/build
/add-longlong-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x0010008b): 11 da 89 45 f0 89 55 f4 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x0010008b is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x0010008b) 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!

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x00100093 QEMU=0x0010008d

(nemu)
```

11(adc)

根据手册填表，把0x10-0x15都填好

```
  /* 0x10 */    IDEXW(G2E, adc, 1), IDEX(G2E, adc), IDEXW(E2G, adc, 1),
IDEX(E2G, adc),
  /* 0x14 */    IDEXW(I2a, adc, 1), IDEX(I2a, adc), EMPTY, EMPTY,
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cputest/build
/add-longlong-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x0010012e): 09 f8 85 c0 0f 94 c0 0f ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x0010012e is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x0010012e) 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!

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x00100136 QEMU=0x00100130

(nemu)
```

09(or)

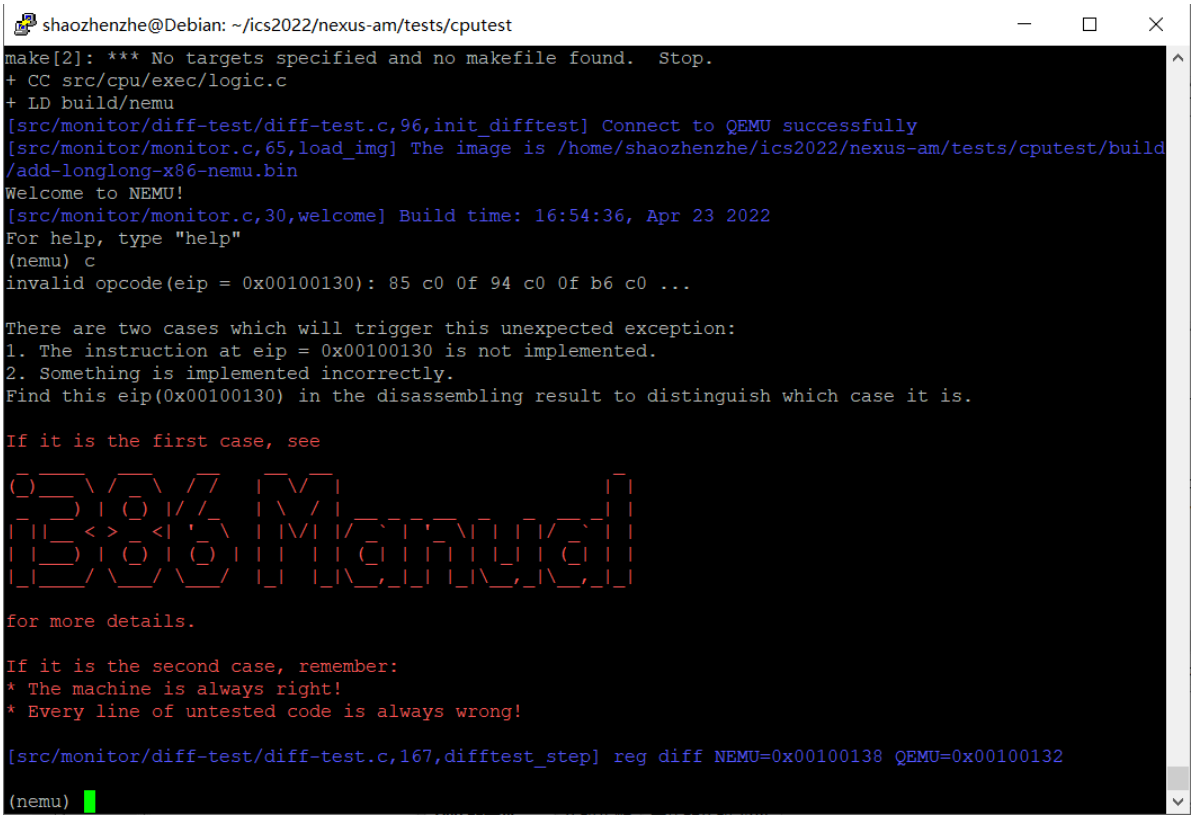
根据手册填表，这里把0x08-0x0d都填好

```
  /* 0x08 */    IDEXW(G2E, or, 1), IDEX(G2E, or), IDEXW(E2G, or, 1), IDEX(E2G,
or),
  /* 0x0c */    IDEXW(I2a, or, 1), IDEX(I2a, or), EMPTY, EX(2byte_esc),
```

完善执行函数`or`，把目的操作数与操作数取或操作，再设置`CF OF`为0，更新`SF ZF`即可

```
make_EHelper(or) {
  //TODO();

  rtl_or(&id_dest->val, &id_dest->val, &id_src->val);
  operand_write(id_dest, &id_dest->val);

  rtl_set_CF(&tzero);//CF<--0
  rtl_set_OF(&tzero);//OF<--0

  rtl_update_ZFSF(&id_dest->val, id_dest->width);

  print_asm_template2(or);
}
```

运行结果



85(test)

把84和85都填了，译码函数`G2E`，执行函数`test`

```
  /* 0x84 */    IDEXW(G2E, test, 1), IDEX(G2E, test), EMPTY, EMPTY,
```

完善`test`执行函数，和与操作相似，区别是`test`只修改标志位，不把结果写入

```
make_EHelper(test) {
  //TODO();
  rtl_and(&id_dest->val, &id_dest->val, &id_src->val);

  rtl_set_CF(&tzero);//CF=0
  rtl_set_OF(&tzero);//OF=0
  rtl_update_ZFSF(&id_dest->val, id_dest->width);//更新ZF、SF

  print_asm_template2(test);
}
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=add-longlong run
Building add-longlong [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests
/cputest/build/add-longlong-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

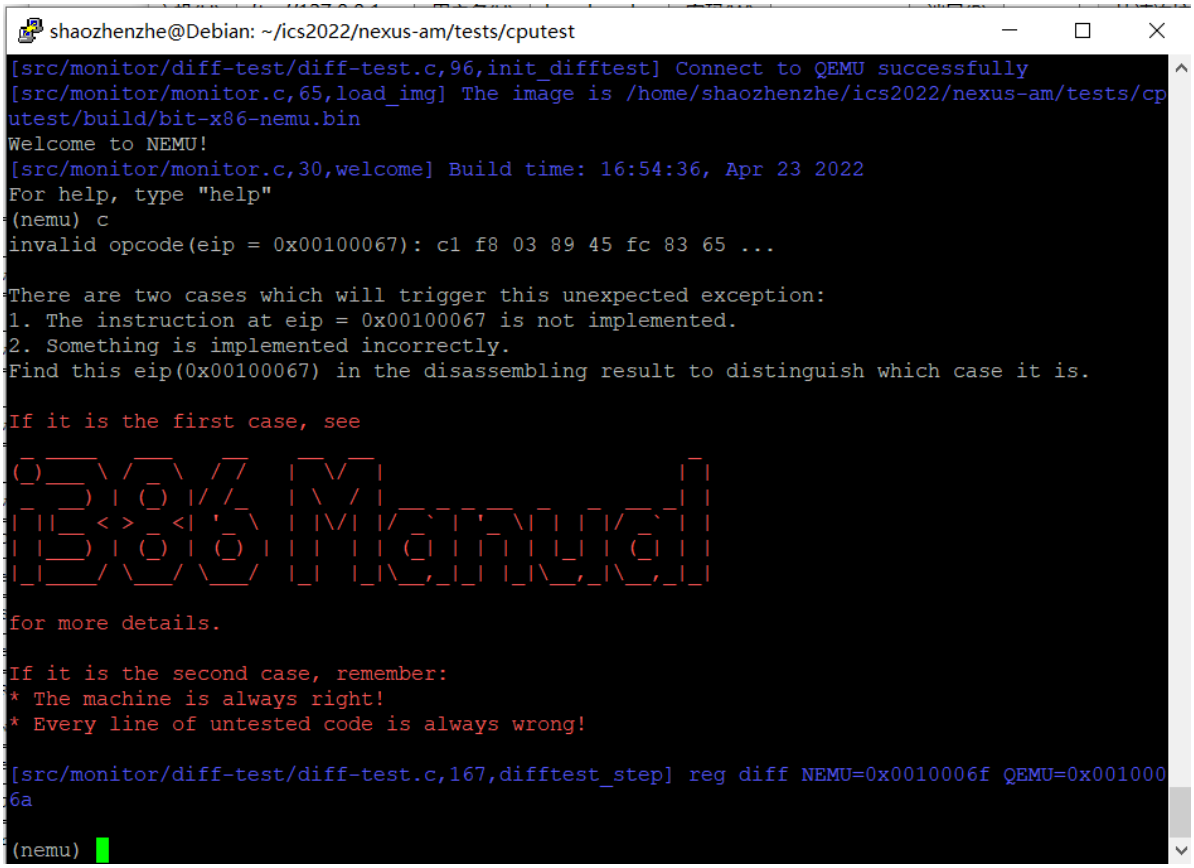
bit.c

6a(push)

译码函数`push_SI`，执行函数`push`，这里把0x68也填好

```
  /* 0x68 */    IDEX(push_SI, push), EMPTY, IDEXW(push_SI, push, 1), EMPTY,
```

运行结果



c1 f8(sar)

查手册得到gp2的`sar`

```
make_group(gp2,
    EMPTY, EMPTY, EMPTY, EMPTY,
    EMPTY, EMPTY, EMPTY, EX(sar))
```

完善`sar`算术右移指令

```
make_EHelper(sar) {
  //TODO();
  // unnecessary to update CF and OF in NEMU

  rtl_sar(&id_dest->val, &id_dest->val, &id_src->val);
  operand_write(id_dest, &id_dest->val);
  rtl_update_ZFSF(&id_dest->val, id_dest->width);

  print_asm_template2(sar);
}
```

运行结果

shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest

```
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/bit-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x0010007b): d3 e2 89 d0 88 45 fb 8b ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x0010007b is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x0010007b) 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!

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x00100083 QEMU=0x001000
7d

(nemu)
```

d3 e2(shl)

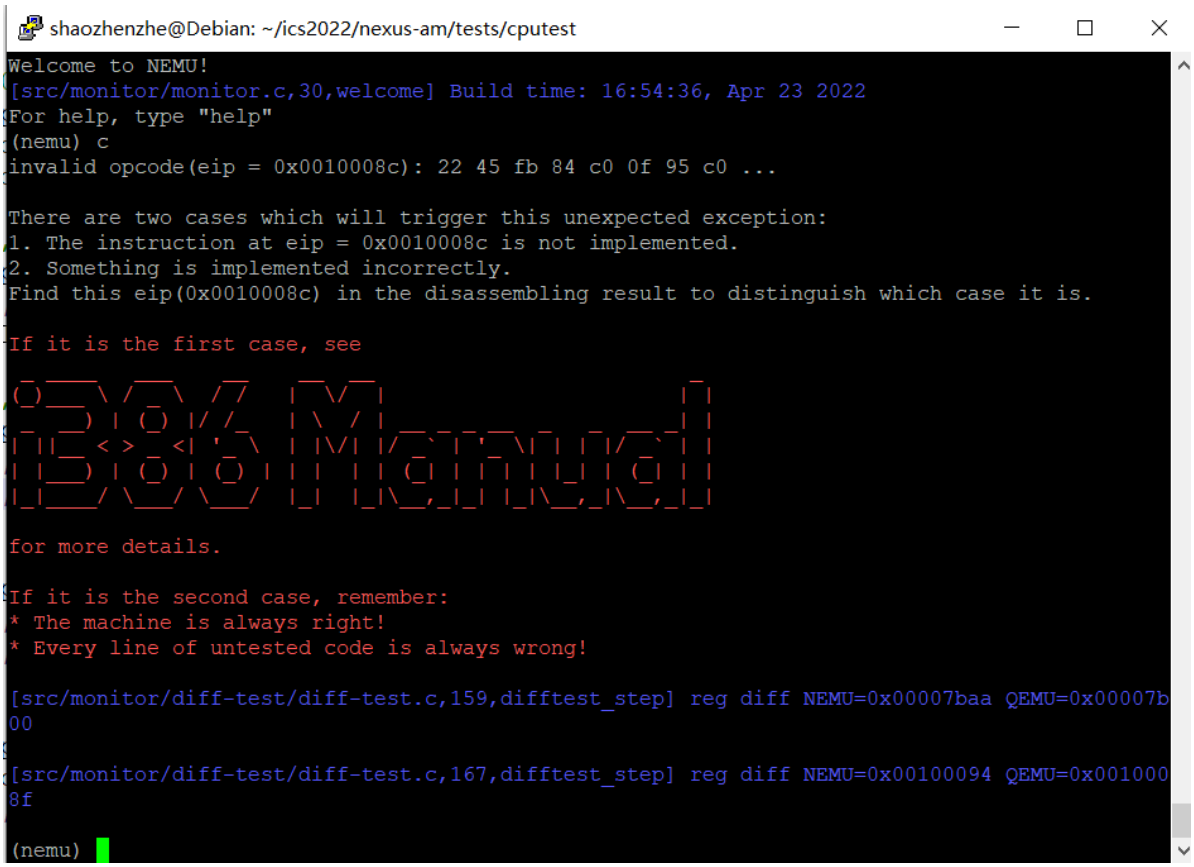
查手册得知是gp2的`shl`，这里顺便把`shr`也填好实现

```
make_group(gp2,
    EMPTY, EMPTY, EMPTY, EMPTY,
    EX(shl), EX(shr), EMPTY, EX(sar))
```

完善`shl`逻辑左移函数和`shr`逻辑右移函数

```
make_EHelper(shl) {
  //TODO();
  // unnecessary to update CF and OF in NEMU

  rtl_shl(&id_dest->val, &id_dest->val, &id_src->val);//逻辑左移
  operand_write(id_dest, &id_dest->val);
  rtl_update_ZFSF(&id_dest->val, id_dest->width);

  print_asm_template2(shl);
}


make_EHelper(shr) {
  //TODO();
  // unnecessary to update CF and OF in NEMU

  rtl_shr(&id_dest->val, &id_dest->val, &id_src->val);//逻辑右移
  operand_write(id_dest, &id_dest->val);
  rtl_update_ZFSF(&id_dest->val, id_dest->width);

  print_asm_template2(shr);
}
```

运行结果



22(and)

查手册，填写0x20-0x25的`and`指令

```
  /* 0x20 */    IDEXW(G2E, and, 1), IDEX(G2E, and), IDEXW(E2G, and, 1),
IDEX(E2G, and),
  /* 0x24 */    IDEXW(I2a, and, 1), IDEX(I2a, and), EMPTY, EMPTY,
```

运行结果

shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest

```
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/bit-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x00100091): 0f 95 c0 c9 c3 55 89 e5 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x00100091 is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x00100091) 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!

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x00100099 QEMU=0x001000
94

(nemu)
```

0f 95(setne)

查第二个表，译码函数`E`，执行函数`setcc`，这里把0x90-0x9f全部填好

```
  /* 0x90 */    IDEXW(E, setcc, 1), IDEXW(E, setcc, 1), IDEXW(E, setcc, 1),
IDEXW(E, setcc, 1),
  /* 0x94 */    IDEXW(E, setcc, 1), IDEXW(E, setcc, 1), IDEXW(E, setcc, 1),
IDEXW(E, setcc, 1),
  /* 0x98 */    IDEXW(E, setcc, 1), IDEXW(E, setcc, 1), IDEXW(E, setcc, 1),
IDEXW(E, setcc, 1),
  /* 0x9c */    IDEXW(E, setcc, 1), IDEXW(E, setcc, 1), IDEXW(E, setcc, 1),
IDEXW(E, setcc, 1),
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x001000db): f7 d0 21 d0 eb 08 8b 45 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x001000db is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x001000db) 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!

[src/monitor/diff-test/diff-test.c,159,difftest_step] reg diff NEMU=0x00007b02 QEMU=0xffff84
fd

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x001000e3 QEMU=0x001000
dd

(nemu)
```

f7 d0(not)

查手册得知是gp3的`not`指令

```
make_group(gp3,
    EMPTY, EMPTY, EX(not), EMPTY,
    EMPTY, EMPTY, EMPTY, EMPTY)
```

完善`not`执行函数，取反即可

```
make_EHelper(not) {
  //TODO();

  rtl_not(&id_dest->val);//取反
  operand_write(id_dest, &id_dest->val);

  print_asm_template1(not);
}
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=bit run
Building bit [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/logic.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/bit-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

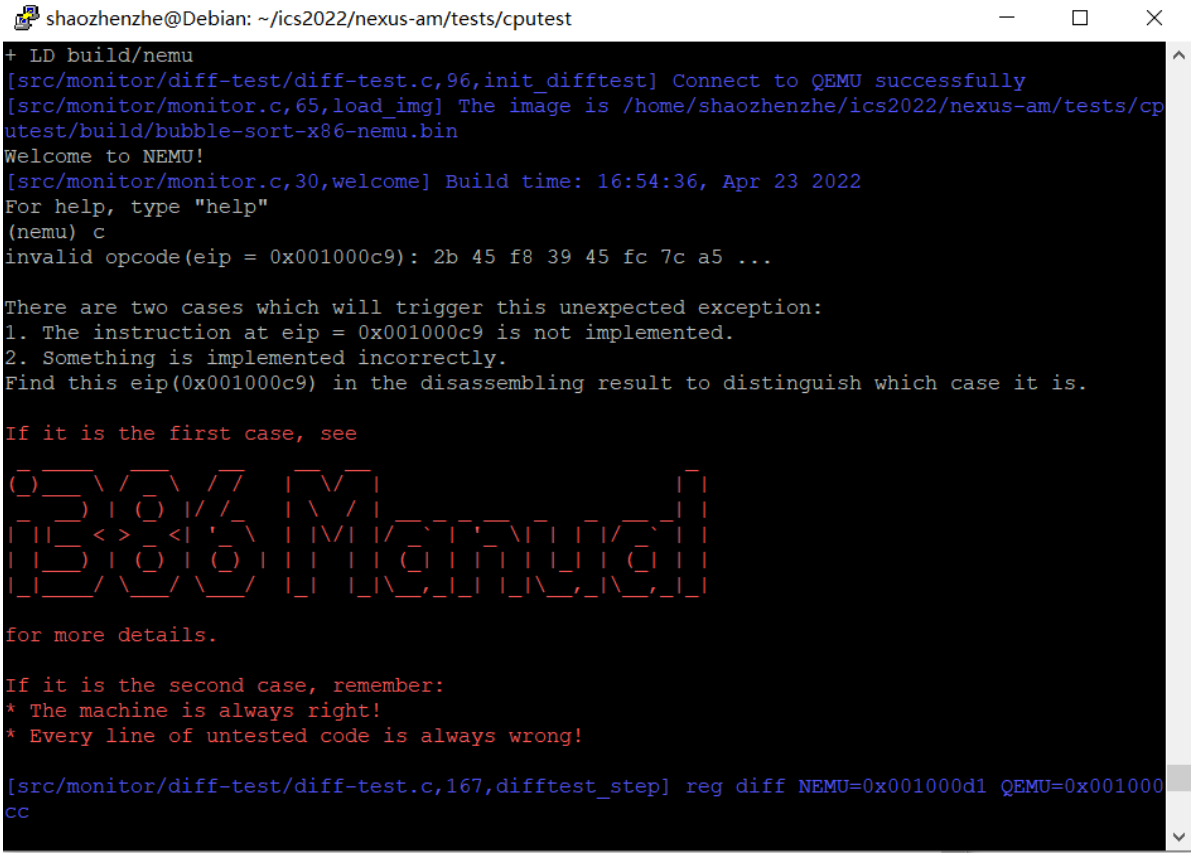
bubble-sort.c

7e(jle)

根据手册填表即可，译码函数`J`，执行函数`jcc`

```
  /* 0x7c */    EMPTY, EMPTY, IDEXW(J, jcc, 1), EMPTY,
```

运行结果



2b(sub)

查手册，填0x28-0x2d，执行函数`sub`在pa2.1已经实现

```
  /* 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,
```

运行结果

shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest

```
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/bubble-sort-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x001000cf): 7c a5 ff 45 f8 83 7d f8 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x001000cf is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x001000cf) 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!

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x001000d7 QEMU=0x001000
76

(nemu) q
```

7c(jl)

和 `7e` 类似

```
/* 0x7c */    IDEXW(J, jcc, 1), EMPTY, IDEXW(J, jcc, 1), EMPTY,
```

运行结果

shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest

```
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x00100083): 40 8b 04 85 c0 01 10 00 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x00100083 is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x00100083) 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!

[src/monitor/diff-test/diff-test.c,159,difftest_step] reg diff NEMU=0x00000000 QEMU=0x000000
01

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x0010008b QEMU=0x001000
84

(nemu)
```

40(inc)

根据手册，填好0x40-0x45

```
  /* 0x40 */    IDEX(r, inc), IDEX(r, inc), IDEX(r, inc), IDEX(r, inc),
  /* 0x44 */    IDEX(r, inc), IDEX(r, inc), IDEX(r, inc), IDEX(r, inc),
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=bubble-sort run
Building bubble-sort [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/bubble-sort-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

dummy.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=dummy run
Building dummy [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/dummy-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

fact.c

74(jz)

和`7c`、`7e`类似，这里顺便把0x70-0x78全部填好

```
  /* 0x70 */    IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), IDEXW(J,
jcc, 1),
  /* 0x74 */    IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), IDEXW(J,
jcc, 1),
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x0010007a): 48 83 ec 0c 50 e8 da ff ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x0010007a is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x0010007a) 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!

[src/monitor/diff-test/diff-test.c,159,difftest_step] reg diff NEMU=0x00000002 QEMU=0x000000
01

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x00100082 QEMU=0x001000
7b

(nemu)
```

48(dec)

根据手册，填0x48-0x4d

```
  /* 0x48 */    IDEX(r, dec), IDEX(r, dec), IDEX(r, dec), IDEX(r, dec),
  /* 0x4c */    IDEX(r, dec), IDEX(r, dec), IDEX(r, dec), IDEX(r, dec),
```

完善`dec`执行函数，模仿sub减1即可

```
make_EHelper(dec) {
  //TODO();

  rtlreg_t temp=1;
  rtl_sub(&t2, &id_dest->val, &temp);//-1
  rtl_sltu(&t3, &t2, &id_dest->val);
  operand_write(id_dest, &t2);

  rtl_update_ZFSF(&t2, id_dest->width);

  rtl_xor(&t0, &id_dest->val, &a);
  rtl_not(&t0);
  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_template1(dec);
}
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x00100087): 0f af 45 08 c9 c3 8d 4c ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x00100087 is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x00100087) 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!

[src/monitor/diff-test/diff-test.c,159,difftest_step] reg diff NEMU=0x00000001 QEMU=0x000000
02

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x0010008f QEMU=0x001000
8b

(nemu)
```

0f af(imul)

查第二个表。译码函数`E2G`，由于是两个操作数，所以执行函数`imul2`

```
/* 0xac */    EMPTY, EMPTY, EMPTY, IDEX(E2G, imul2),
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=fact run
Building fact [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/fact-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

fib.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=fib run
Building fib [x86-nemu]
+ CC tests/fib.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/fib-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

goldbach.c

7f(jnle)

和 `7c` 、`7e` 类似，这里顺便把0x78-0x7f全部填好

```
  /* 0x78 */    IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), IDEXW(J,
jcc, 1),
  /* 0x7c */    IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), IDEXW(J, jcc, 1), IDEXW(J,
jcc, 1),
```

运行结果

shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest

```
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x0010007d): 99 f7 7d fc 89 d0 85 c0 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x0010007d is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x0010007d) 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!

[src/monitor/diff-test/diff-test.c,161,difftest_step] reg diff NEMU=0x1ec38819 QEMU=0x000000
00

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x00100085 QEMU=0x001000
7e

(nemu)
```

99(cltd)

无译码函数，填表，这里顺便完成**98(cwtl)**指令

```
  /* 0x98 */    EX(cwtl), EX(cltd), EMPTY, EMPTY,
```

完善执行函数 `cltd`

当操作数为16位时，若 `ax<0` 则给 `dx` 赋值 `0xffff`，否则赋值 `0`

当操作数为32位时，若`eax<0`则给`edx`赋值`0xffffffff`否则赋值`0`

```
make_EHelper(cltd) {
  if (decoding.is_operand_size_16) {
    //TODO();
    if ((int16_t)(cpu.eax&0xffff)<0) {//ax<0
       cpu.edx=0xffff;
    }
    else {
       cpu.edx=0;
    }
  }
  else {
    //TODO();
    if ((int32_t)(cpu.eax)<0) {//eax<0
       cpu.edx=0xffffffff;
     }
     else {
       cpu.edx=0;
     }
  }

  print_asm(decoding.is_operand_size_16 ? "cwtl" : "cltd");
}
```

完善`cwtl`执行函数

也是分情况完成对寄存器的符号扩展

```
make_EHelper(cwtl) {
  if (decoding.is_operand_size_16) {
    //TODO();
    rtl_lr_b(&t0, R_AX);
    rtl_sext(&t0, &t0, 1);
    rtl_sr_w(R_AX, &t0);
  }
  else {
    //TODO();
    rtl_lr_w(&t0, R_AX);
    rtl_sext(&t0, &t0, 2);
    rtl_sr_l(R_EAX, &t0);
  }

  print_asm(decoding.is_operand_size_16 ? "cbtw" : "cwtl");
}
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/tests/cputest
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
invalid opcode(eip = 0x0010007e): f7 7d fc 89 d0 85 c0 75 ...

There are two cases which will trigger this unexpected exception:
1. The instruction at eip = 0x0010007e is not implemented.
2. Something is implemented incorrectly.
Find this eip(0x0010007e) 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!

[src/monitor/diff-test/diff-test.c,159,difftest_step] reg diff NEMU=0x00000004 QEMU=0x000000
02

[src/monitor/diff-test/diff-test.c,167,difftest_step] reg diff NEMU=0x00100086 QEMU=0x001000
81

(nemu)
```

f7 7d(idiv)

查手册得知是gp3的`idiv`

```
make_group(gp3,
    EMPTY, EMPTY, EX(not), EMPTY,
    EMPTY, EMPTY, EMPTY, EX(idiv))
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=goldbach run
Building goldbach [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/goldbach-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

if-else.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=if-else run
Building if-else [x86-nemu]
+ CC tests/if-else.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/if-else-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

leap-year.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=leap-year run
Building leap-year [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/leap-year-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

load-store.c

0f bf(movsx)

查第二个表得到是`movsx`，这里把0xbe也填好

```
/* 0xbc */    EMPTY, EMPTY, IDEXW(E2G, movsx, 1), IDEXW(E2G, movsx, 2),
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=load-store run
Building load-store [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/load-store-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

matrix-mul.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=matrix-mul run
Building matrix-mul [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/matrix-mul-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

max.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=max run
Building max [x86-nemu]
+ CC tests/max.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/max-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

min3.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=min3 run
Building min3 [x86-nemu]
+ CC tests/min3.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/min3-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

mov-c.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=mov-c run
Building mov-c [x86-nemu]
+ CC tests/mov-c.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/mov-c-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

movsx.c

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=movsx run
Building movsx [x86-nemu]
+ CC tests/movsx.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/movsx-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

mul-longlong.c

f7 65(mul)

查手册得到是gp3的`mul`

```
make_group(gp3,
    EMPTY, EMPTY, EX(not), EMPTY,
    EX(mul), EMPTY, EMPTY, EX(idiv))
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=mul-longlong run
Building mul-longlong [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/mul-longlong-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

pascal.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=pascal run
Building pascal [x86-nemu]
+ CC tests/pascal.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/pascal-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

prime.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=prime run
Building prime [x86-nemu]
+ CC tests/prime.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/prime-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

quick-sort.c

ff 4d(dec)

查手册得到是gp5的`dec`

```
make_group(gp5,
    EX(inc), EX(dec), EMPTY, EMPTY,
    EMPTY, EMPTY, EX(push), EMPTY)
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=quick-sort run
Building quick-sort [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/cp
utest/build/quick-sort-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

recursion.c

ff d0(call)

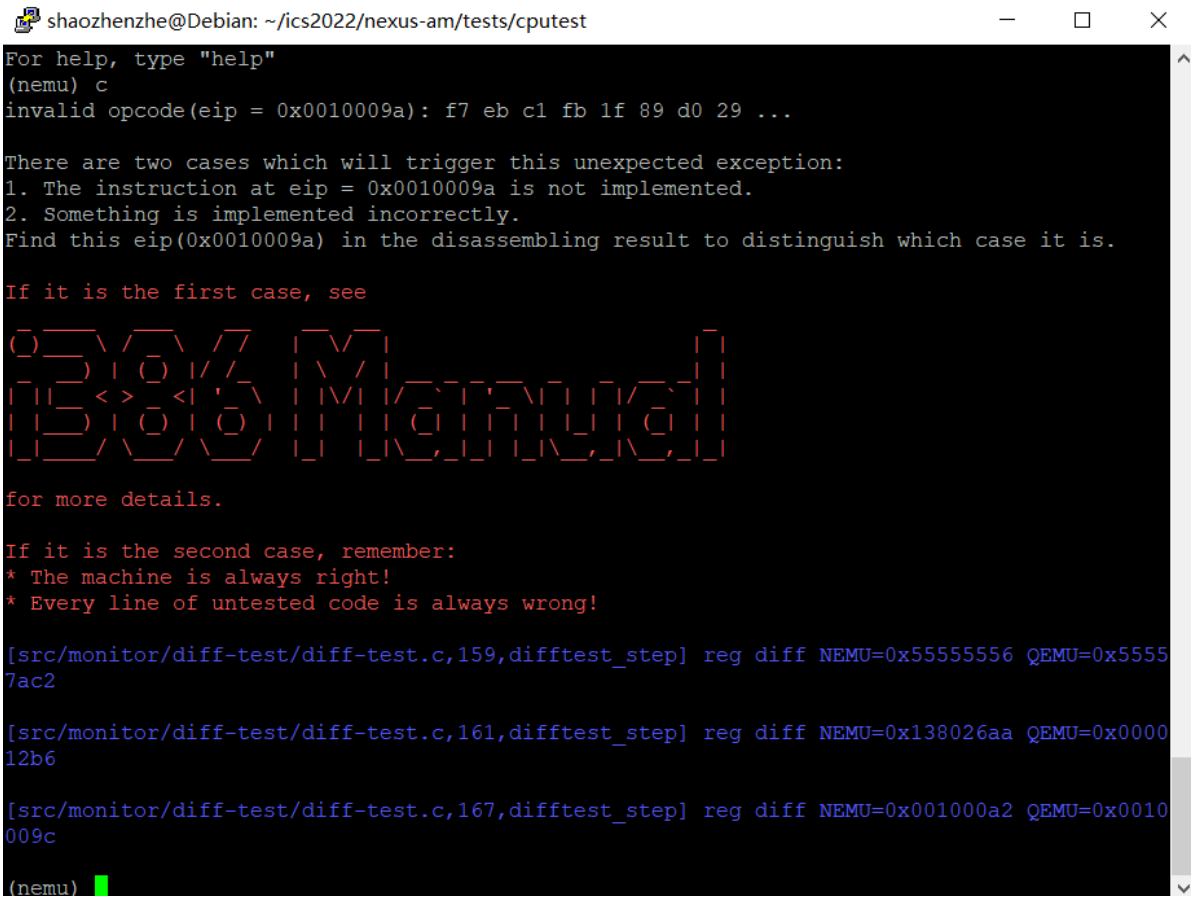
查手册得到gp5的`call`，执行函数是`call_rm`

```
make_group(gp5,
    EX(inc), EX(dec), EX(call_rm), EMPTY,
    EMPTY, EMPTY, EX(push), EMPTY)
```

完善执行函数`call_rm`，首先给`is_jmp`赋值1，然后给`jmp_eip`赋值`id_dest->val`，最后`push eip`

```
make_EHelper(call_rm) {
  //TODO();

  decoding.is_jmp = 1;
  decoding.jmp_eip = id_dest->val;
  rtl_push(eip);

  print_asm("call *%s", id_dest->str);
}
```

运行结果



f7 eb(imul)

查手册得到gp3的imul，执行函数是imul1

```
make_group(gp3,
    EMPTY, EMPTY, EX(not), EMPTY,
    EX(mul), EX(imul1), EMPTY, EX(idiv))
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=recursion run
Building recursion [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ CC src/cpu/exec/arith.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/recursion-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

select-sort.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=select-sort run
Building select-sort [x86-nemu]
+ CC tests/select-sort.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/select-sort-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

shift.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=shift run
Building shift [x86-nemu]
+ CC tests/shift.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/shift-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

shuixianhua.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=shuixianhua run
Building shuixianhua [x86-nemu]
+ CC tests/shuixianhua.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/shuixianhua-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

sub-longlong.c

1b(sbb)

查表把0x18-0x1d填完

```
  /* 0x18 */    IDEXW(G2E, sbb, 1), IDEX(G2E, sbb), IDEXW(E2G, sbb, 1),
IDEX(E2G, sbb),
  /* 0x1c */    IDEXW(I2a, sbb, 1), IDEX(I2a, sbb), EMPTY, EMPTY,
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=sub-longlong run
Building sub-longlong [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/sub-longlong-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

sum.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=sum run
Building sum [x86-nemu]
+ CC tests/sum.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/sum-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

switch.c

ff e0(jmp)

查手册得知是gp5的jmp，执行函数是jmp_rm

```
make_group(gp5,
    EX(inc), EX(dec), EX(call_rm), EMPTY,
    EX(jmp_rm), EMPTY, EX(push), EMPTY)
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=switch run
Building switch [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/switch-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

to-lower-case.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=to-lower-case run
Building to-lower-case [x86-nemu]
+ CC tests/to-lower-case.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/to-lower-case-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

unalign.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=unalign run
Building unalign [x86-nemu]
+ CC tests/unalign.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/unalign-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

wanshu.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=wanshu run
Building wanshu [x86-nemu]
+ CC tests/wanshu.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/wanshu-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

string.c

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=string run
Building string [x86-nemu]
+ CC tests/string.c
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/string-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010001b

(nemu)
```

hello-str

f7 75(div)

查手册得知是gp3的div

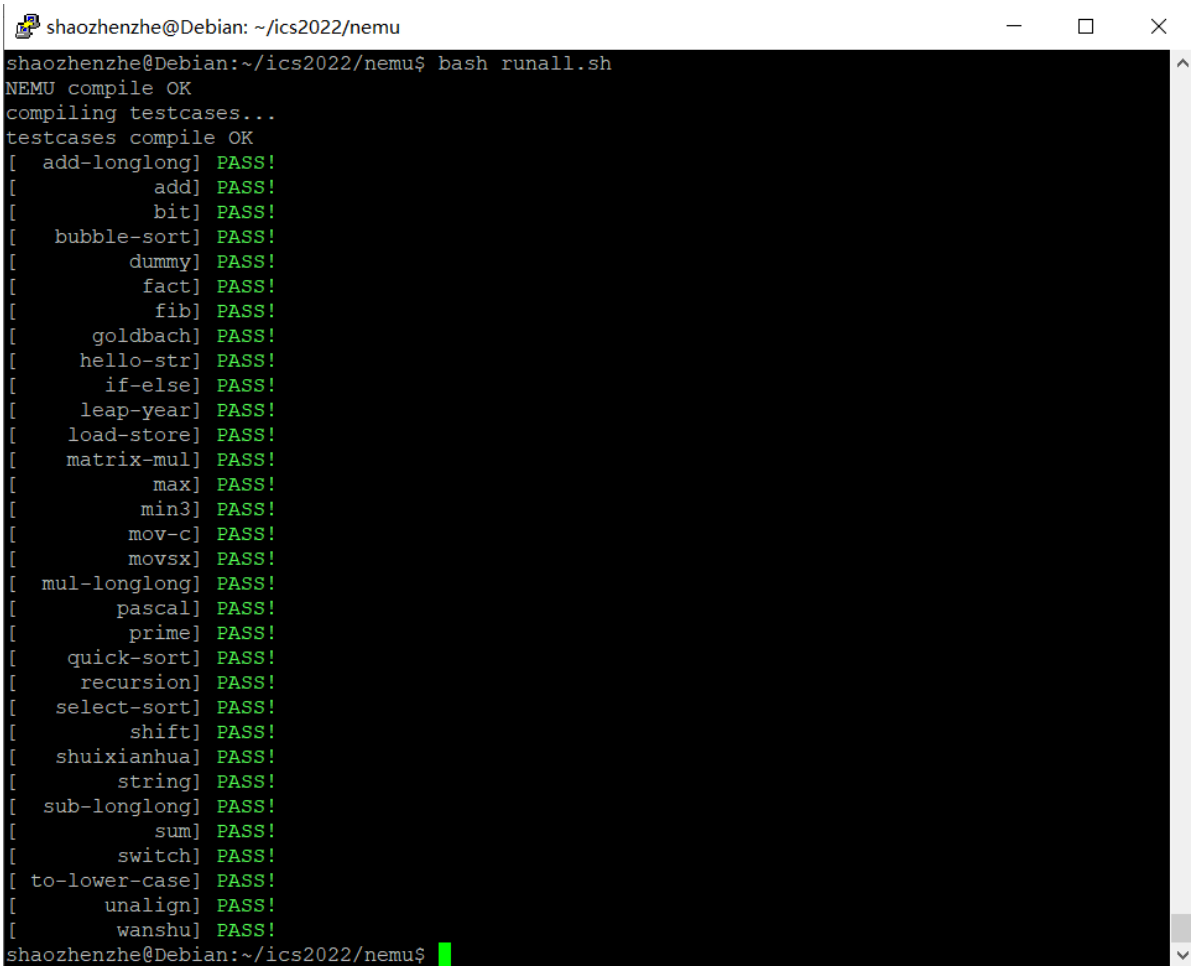
```
make_group(gp3,
    EMPTY, EMPTY, EX(not), EMPTY,
    EX(mul), EX(imul1), EX(div), EX(idiv))
```

运行结果

```
shaozhenzhe@Debian:~/ics2022/nexus-am/tests/cputest$ make ALL=hello-str run
Building hello-str [x86-nemu]
Building am [x86-nemu]
make[2]: *** No targets specified and no makefile found.  Stop.
+ CC src/cpu/exec/exec.c
+ LD build/nemu
[src/monitor/diff-test/diff-test.c,96,init_difftest] Connect to QEMU successfully
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/tests/
cputest/build/hello-str-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 16:54:36, Apr 23 2022
For help, type "help"
(nemu) c
nemu: HIT GOOD TRAP at eip = 0x0010002a

(nemu)
```

通过一键回归测试（5 分）

- 需看到全部测试样例通过的截图。



IN/OUT 指令（10 分）

- 实现 IN , OUT 两条指令；
- 成功运行 nexus-am/apps/hello 程序。

在 `nexus-am/am/arch/x86-nemu/src/trm.c` 中定义宏 `HAS_SERIAL`

在 `nemu/include/common.h` 中定义宏 `HAS_IOE`

先查手册填表

```
  /* 0xe4 */    IDEXW(in_I2a, in, 1), IDEX(in_I2a, in), IDEXW(out_a2I, out, 1),
IDEX(out_a2I, out),
  /* 0xec */    IDEXW(in_dx2a, in, 1), IDEX(in_dx2a, in), IDEXW(out_a2dx, out,
1), IDEX(out_a2dx, out),
```

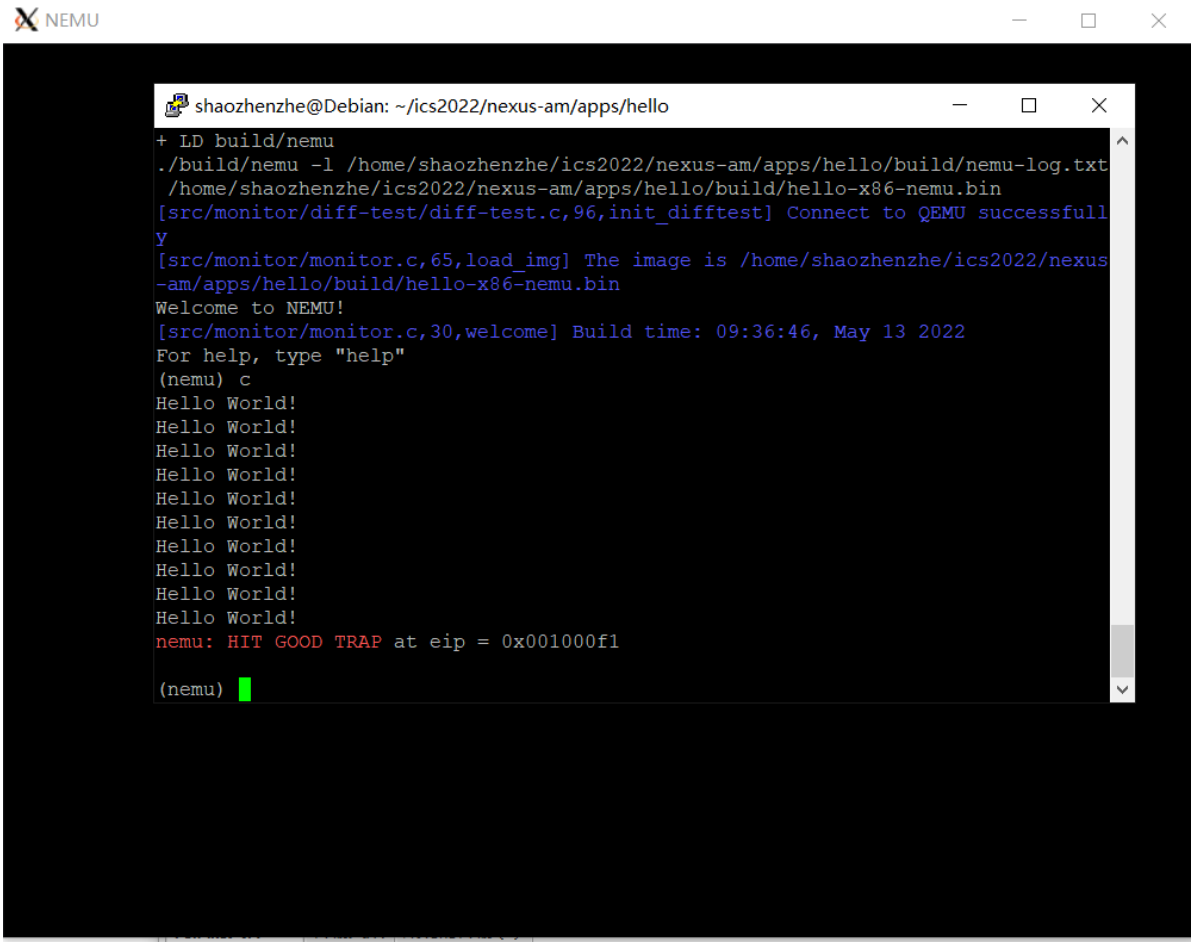
完善执行函数in和out，通过 `pio_read` 读取指定位置的值并写入目的操作数即可

`out`：通过 `pio_write` 读取指定位置的值输出即可

```
make_EHelper(in) {
  //TODO();
  id_dest->val=pio_read(id_src->val, id_dest->width);
  operand_write(id_dest, &id_dest->val);

  print_asm_template2(in);

#ifdef DIFF_TEST
```

```
  diff_test_skip_qemu();
#endif
}

make_EHelper(out) {
  //TODO();
  pio_write(id_dest->val, id_dest->width, id_src->val);

  print_asm_template2(out);

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

运行结果



实现时钟设备（10 分）

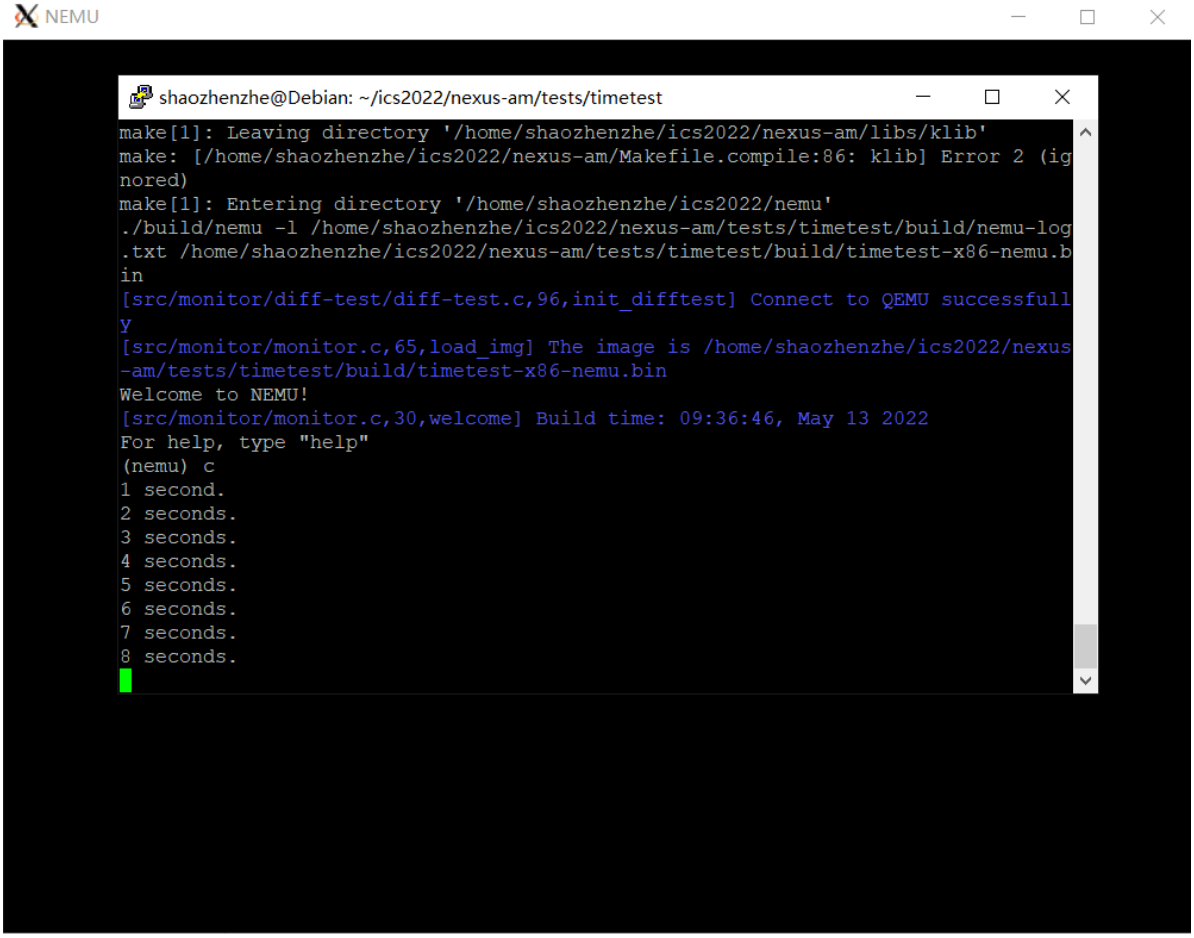
- 实现 _uptime() 函数;
- 成功运行 timetest 程序。

`nexus-am/am/arch/x86-nemu/src/ioe.c`

获取当前时间减去初始时间(boot_time)即可

```
unsigned long _uptime() {
  return inl(RTC_PORT)-boot_time;
}
```

运行结果



运行跑分项目（10 分）

- 成功运行 dhrystone , coremark , microbench 三个跑分项目；
- 其中可能需要实现一个 rol 指令。

1.dhrystone

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/apps/dhrystone
+ CC src/cpu/exec/arith.c
+ CC src/cpu/reg.c
+ CC src/cpu/intr.c
+ CC src/cpu/decode/decode.c
+ CC src/cpu/decode/modrm.c
+ CC src/monitor/debug/ui.c
+ CC src/monitor/debug/watchpoint.c
+ CC src/monitor/debug/expr.c
+ CC src/monitor/monitor.c
+ CC src/monitor/diff-test/gdb-host.c
+ CC src/monitor/diff-test/diff-test.c
+ CC src/monitor/diff-test/protocol.c
+ CC src/monitor/cpu-exec.c
+ LD build/nemu
./build/nemu -l /home/shaozhenzhe/ics2022/nexus-am/apps/dhrystone/build/nemu-log.txt /home
/shaozhenzhe/ics2022/nexus-am/apps/dhrystone/build/dhrystone-x86-nemu.bin
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/apps/d
hrystone/build/dhrystone-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 09:52:08, May 13 2022
For help, type "help"
(nemu) c
Dhrystone Benchmark, Version C, Version 2.2
Trying 500000 runs through Dhrystone.
Finished in 49641 ms
==================================================
Dhrystone PASS         20 Marks
                   vs. 100000 Marks (i7-6700 @ 3.40GHz)
nemu: HIT GOOD TRAP at eip = 0x001000f1

(nemu)
```

2.coremark

需要实现f7 d8(neg)

查手册得知是gp3的neg

```
make_group(gp3,
    EMPTY, EMPTY, EX(not), EX(neg),
    EX(mul), EX(imul1), EX(div), EX(idiv))
```

完善neg执行函数，即取相反数，并更新标志位(模仿adc即可)

```
make_EHelper(neg) {
  //TODO();

  if(id_dest->val==0){
    t0=0;
    rtl_set_CF(&t0);
  }else{
    t0=1;
    rtl_set_CF(&t0);
  }

  id_dest->val=-id_dest->val;//取相反数
  operand_write(id_dest, &id_dest->val);
  rtl_update_ZFSF(&id_dest->val, id_dest->width); //更新ZF SF
  //更新OF
  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_template1(neg);
}
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/apps/coremark
make[1]: Leaving directory '/home/shaozhenzhe/ics2022/nexus-am/libs/klib'
make: [/home/shaozhenzhe/ics2022/nexus-am/Makefile.compile:86: klib] Error 2 (ignored)
make[1]: Entering directory '/home/shaozhenzhe/ics2022/nemu'
+ CC src/cpu/exec/arith.c
+ LD build/nemu
./build/nemu -l /home/shaozhenzhe/ics2022/nexus-am/apps/coremark/build/nemu-log.txt /home/
shaozhenzhe/ics2022/nexus-am/apps/coremark/build/coremark-x86-nemu.bin
[src/monitor/monitor.c,65,load_img] The image is /home/shaozhenzhe/ics2022/nexus-am/apps/c
oremark/build/coremark-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 09:52:08, May 13 2022
For help, type "help"
(nemu) c
Running CoreMark for 1000 iterations
2K performance run parameters for coremark.
CoreMark Size    : 666
Total time (ms)  : 109620
Iterations       : 1000
Compiler version : GCC8.3.0
seedcrc          : 0xe9f5
[0]crclist       : 0xe714
[0]crcmatrix     : 0x1fd7
[0]crcstate      : 0x8e3a
[0]crcfinal      : 0xd340
Finised in 109620 ms.
==================================================
CoreMark PASS       40 Marks
               vs. 100000 Marks (i7-6700 @ 3.40GHz)
nemu: HIT GOOD TRAP at eip = 0x001000f1

(nemu)
```

3.microbench

需要实现**d3 c2(rol)**指令

```
make_group(gp2,
    EX(rol), EMPTY, EMPTY, EMPTY,
    EX(shl), EX(shr), EMPTY, EX(sar))
```

完善执行函数rol，完成循环左移操作，CF标志位根据最后一个移出去的位来设置

```
make_EHelper(rol) {
  //TODO();
  for (t0=0;t0<id_src->val;t0++) {
    rtl_shri(&t1, &id_dest->val, id_dest->width*8-1);//获取最高位
    rtl_shli(&t2, &id_dest->val, 1);//左移一位
    id_dest->val=t1+t2;//把最高位补在最低位，实现循环左移
  }
  rtl_set_CF(&t1);//设置CF
  operand_write(id_dest, &id_dest->val);

  print_asm_template2(rol);
}
```

运行结果

```
shaozhenzhe@Debian: ~/ics2022/nexus-am/apps/microbench
icrobench/build/microbench-x86-nemu.bin
Welcome to NEMU!
[src/monitor/monitor.c,30,welcome] Build time: 09:52:08, May 13 2022
For help, type "help"
(nemu) c
[qsort] Quick sort: * Passed.
  min time: 6519 ms [84]
[queen] Queen placement: * Passed.
  min time: 4092 ms [126]
[bf] Brainf**k interpreter: * Passed.
  min time: 42318 ms [61]
[fib] Fibonacci number: * Passed.
  min time: 403675 ms [7]
[sieve] Eratosthenes sieve: * Passed.
  min time: 139993 ms [30]
[15pz] A* 15-puzzle search: * Passed.
  min time: 23809 ms [24]
[dinic] Dinic's maxflow algorithm: * Passed.
  min time: 12130 ms [111]
[lzip] Lzip compression: * Passed.
  min time: 72622 ms [36]
[ssort] Suffix sort: * Passed.
  min time: 7707 ms [76]
[md5] MD5 digest: * Passed.
  min time: 62403 ms [31]
==================================================
MicroBench PASS        58 Marks
                  vs. 100000 Marks (i7-6700 @ 3.40GHz)
nemu: HIT GOOD TRAP at eip = 0x001000f1

(nemu)
```

实现键盘设备（10 分）

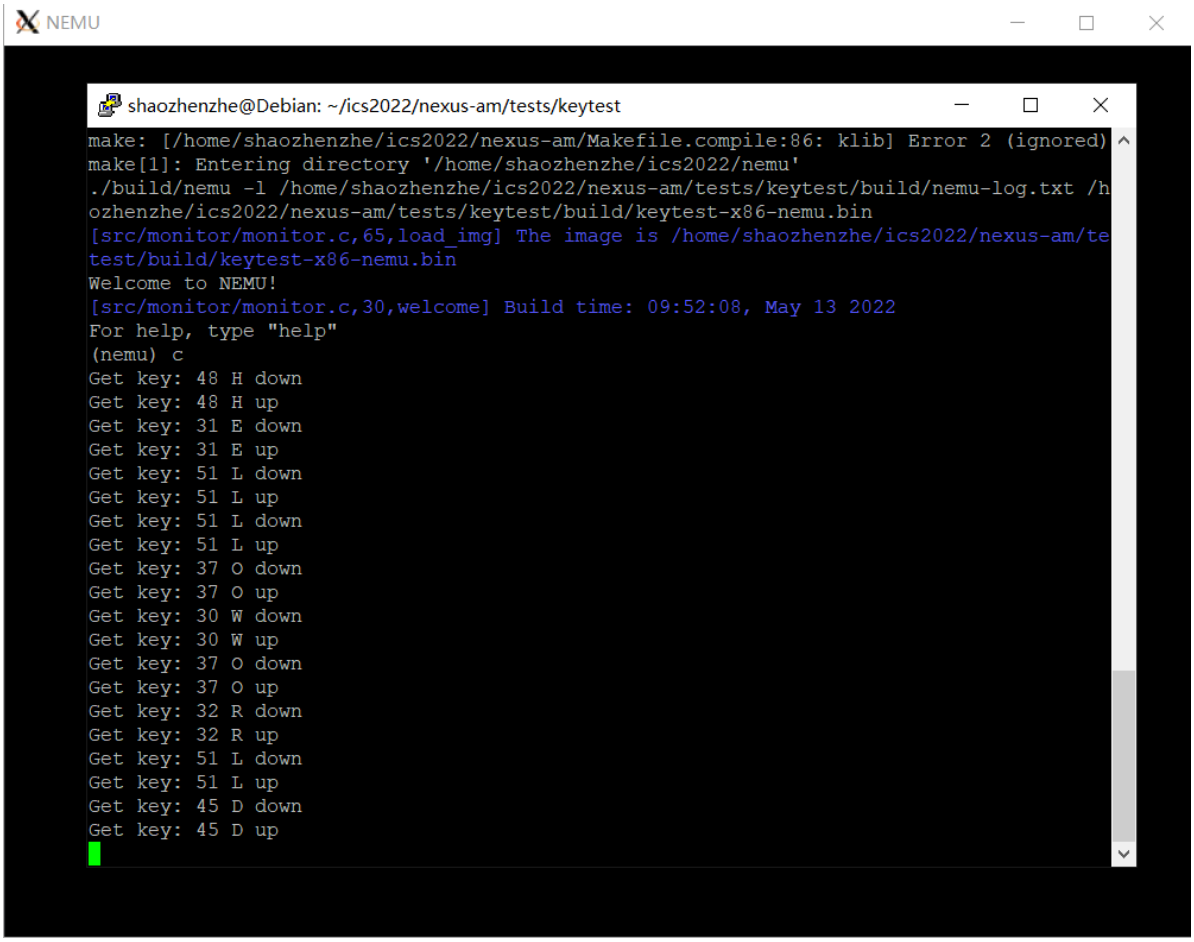
- 实现 _read_key() 函数；
- 成功运行 keytest 程序。

`nexus-am/am/arch/x86-nemu/src/ioe.c`

根据讲义的说明，可以先读取 `0x64` 判断是否有键盘码放入数据寄存器，如果有就去 `0x60` 端口取数据

```
int _read_key() {
  uint32_t make_code=_KEY_NONE;
  if (inb(0x64)) {  //读取0x64处的一个字节，判断状态
    make_code=inl(0x60);//读取0x60处的数据
  }

  return make_code;
}
```

运行结果



添加内存映射 I/O （10 分）

- 在 paddr_read() 和 paddr_write() 中添加内存映射 I/O 判断；
- 成功运行 videotest 程序

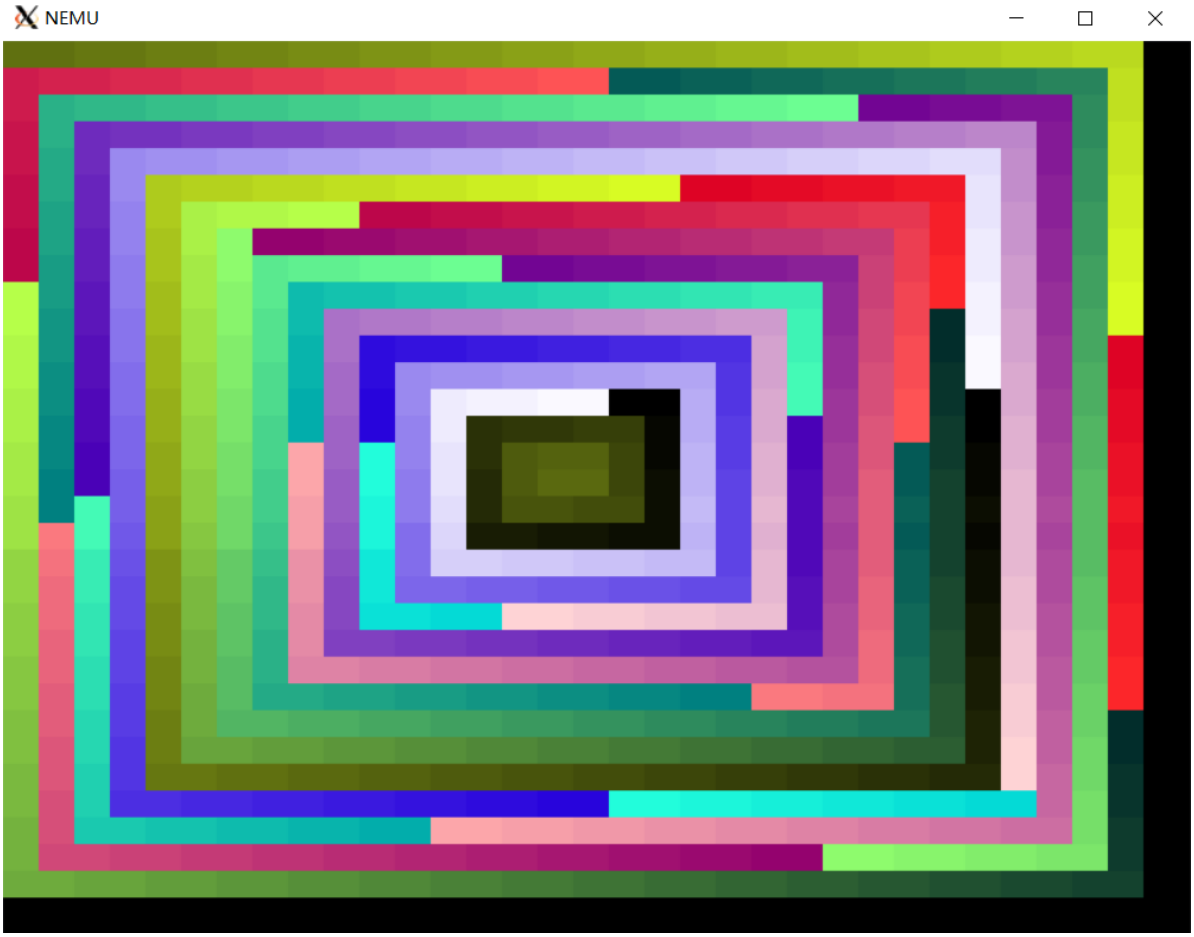
`nemu/src/memory/memory.c`

根据讲义，通过 `is_mmio()` 函数判断一个物理地址是否被映射到 I/O 空间, 如果是, `is_mmio()` 会返回映射号, 否则返回 `-1`. 内存映射 I/O 的访问需要调用 `mmio_read()` 或 `mmio_write()`, 调用时需要提供映射号. 如果不是内存映射 I/O 的访问, 就访问 `pmem`.

```
uint32_t paddr_read(paddr_t addr, int len) {
  int mmio_id=is_mmio(addr);
  if(mmio_id != -1){
    return mmio_read(addr, len, mmio_id);
  }
  else{
    return pmem_rw(addr, uint32_t) & (~0u >> ((4 - len) << 3));
  }
}


void paddr_write(paddr_t addr, int len, uint32_t data) {
  int mmio_id=is_mmio(addr);
  if(mmio_id != -1){
    return mmio_write(addr, len, data, mmio_id);
  }
  else{
    memcpy(guest_to_host(addr), &data, len);
```

```
    }
  }
```

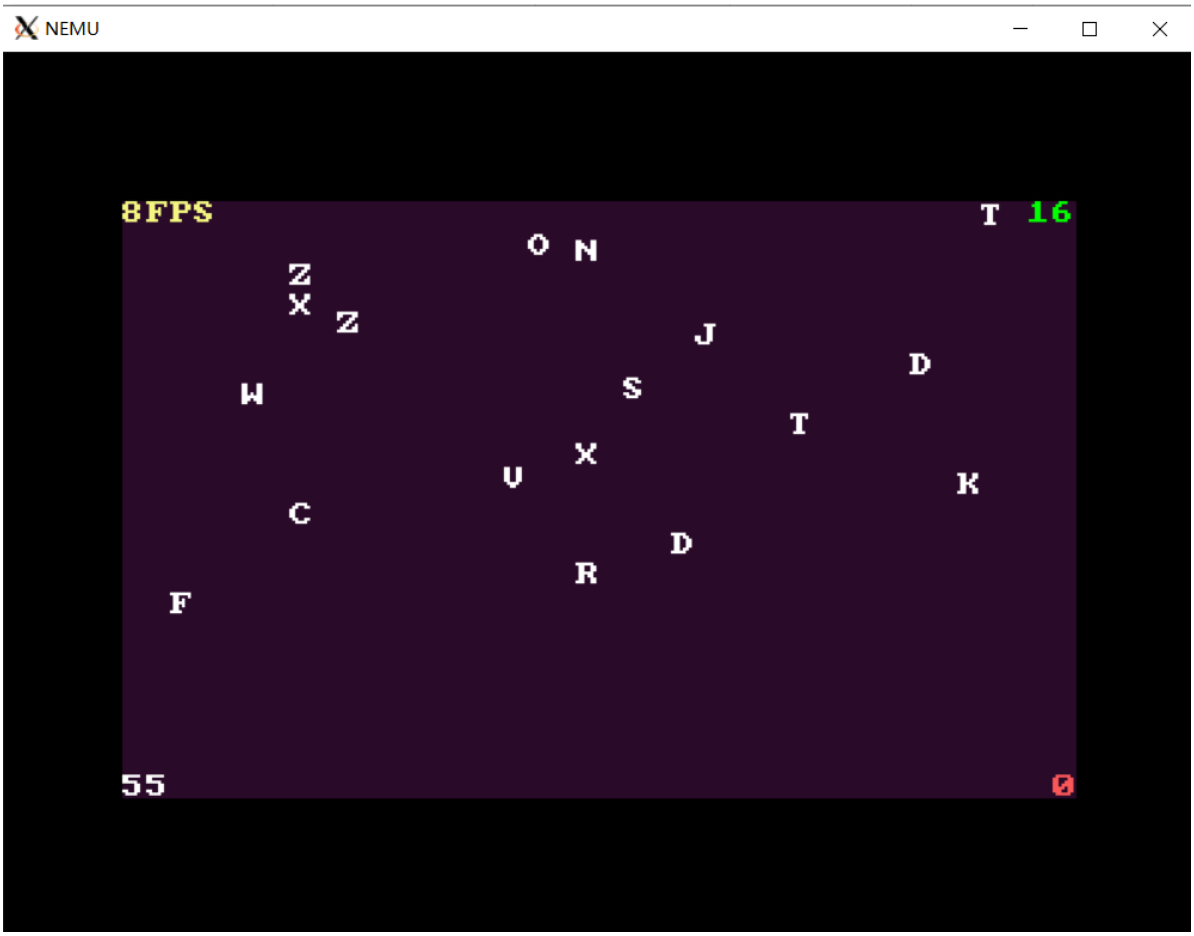
运行结果



运行打字小游戏（5 分）

- 帧数（FPS）不低于 3 。

运行结果



遇到的问题及解决办法

1.问题：在实现跑测试样例`add-longlong`的过程中，出现了`diff-test`寄存器值不同的错误

解决方法：通过查看反汇编发现是`xor`指令错误，`xor`指令是在pa2.1实现的，仔细检查后发现实现错误，pa2.1确实没有对`xor`检验的部分，因此这里出现错误了。改为以下即可。

```
make_EHelper(xor) {
  //TODO();
  rtl_xor(&id_dest->val,&id_dest->val, &id_src->val);//异或
  operand_write(id_dest,&id_dest->val);//赋值

  rtl_set_CF(&tzero);//CF=0
  rtl_set_OF(&tzero);//OF=0
  rtl_update_ZFSF(&id_dest->val,id_dest->width);//更新ZF、SF

  print_asm_template2(xor);
}
```

2.问题：在测试`microbench`跑分项目时，出现了`physical address(0xfffffffc) is out of bond`错误

解决办法：`common.h`打开`DEBUG`和`DIFF_TEST`的宏，运行到错误的地方时查看反汇编结果，发现是`c2(ret)`指令错误。通过群里同学的帮助，得知`ret`需要在`0xc2`处特判。把`ret`函数改为以下即可

```
make_EHelper(ret) {
  //TODO();
  decoding.is_jmp = 1;
  rtl_pop(&decoding.jmp_eip);
  if(decoding.opcode==0xc2) //0xc2处，esp+dest
  {
     cpu.esp+=id_dest->val;
  }
  print_asm("ret");
}
```

实验心得

本次实验内容很多，实现了很多指令，实现了输入输出，最终能够成功运行打字小游戏，成就感很足。我对于I/O与CPU是如何交互有了更深的了解，理解并实现了内存映射，完成了读取键盘输入的任务。这次实验也让我明白了需要谨慎和仔细，一步一步扎实完成，比如PA2.1中我的ret指令没有实现好，结果在这次实验中给我带来了很大的麻烦。总而言之，本次PA2.2&2.3让我更加深入地理解了计算机的各个功能，收获很大。

其他备注

无