LAB2.1实验报告 张佳豪 2200013093

框架说明

项目构建

根Makefile

将`make T=name`分发到子目录分别构建Simulator和Test. 将T=name传递到test构建对应的测试用例

最后使用simulator执行对应的构建好的测试(.bin)

Simulator

使用Makefile将所有的C源文件编译为目标文件(.o),然后链接所有目标文件为可执行文件simulator

Test

接收根Makefile传递的TARGET，推导依赖关系构建出对应的目标文件(.o), 其中除了target对应对的目标文件还有start.o和trm.o作为通用的abi接口，将三个目标文件依据linker.ld链接为ELF,最后再利用工具生成对应的二进制文件(.bin),并提供易于检查的文本格式的汇编代码(.txt).

模拟器执行逻辑

模拟器使用指令级模拟，将riscv体系结构的程序员可见状态保存在数据结构(`CPU_state`, `mem`)中，如pc、寄存器、内存等

模拟器每次依据pc从指令内存中取出指令，解码并依据指令修改程序员可见状态，更新pc，直到遇到结束指令或异常指令。

其中解码部分采用了嵌套宏构建的字符串匹配器，依次从上到下匹配指令类型并执行第一个匹配的指令的操作。

功能测试与报告

基础功能

通过了基础功能测试用例. 运行:

```
./driver
```

输出:

```
Checking Denpendencies...
Build Simulator...
Simulator building finished.
Processing: ackermann
Success
Processing: add
```

```
Success
Processing: div
Success
Processing: dummy
Success
Processing: if-else
Success
Processing: load-store
Success
Processing: matrix-mul
Success
Processing: puts
Success
Processing: quicksort
Success
Processing: shift
Success
Processing: unalign
Success
Score: 11/11
```

2.Debug支持

运行:

```
make T=dummy MODE=debug
```

输出:

```
-------Build Simulator-------
make[1]: Entering directory '/workspaces/Workspace/lab2_1/simulator/sim'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/workspaces/Workspace/lab2_1/simulator/sim'
-------Build Test-------
make[1]: Entering directory '/workspaces/Workspace/lab2_1/simulator/test'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/workspaces/Workspace/lab2_1/simulator/test'
-------Start Simulation-------
[INFO] (src/memory.c:74) Physical Memory Range:[0000000008000000,
000000000fffffff].
[INFO] (src/memory.c:80) The image is test/build/dummy.bin, size = 176.
help: print this help message
c: continue the stopped program
q: exit the simulator
si [N]: single step N times (default 1)
info r: print register status
b ADDR(Hex): set a breakpoint at ADDR
d: delete all breakpointsx N ADDR(Hex): print 4N bytes at ADDR of the memory. Show
in little endian
>
```

可以按照help中的提示进行调试，除了要求外还实现了断点功能配合continue.

附加功能

断点功能

输出

```
[INFO] (src/memory.c:74) Physical Memory Range:[0000000008000000,
000000000fffffff].
[INFO] (src/memory.c:80) The image is test/build/load-store.bin, size = 936.
help: print this help message
c: continue the stopped program
q: exit the simulator
si [N]: single step N times (default 1)
info r: print register status
b ADDR(Hex): set a breakpoint at ADDR
d: delete all breakpointsx N ADDR(Hex): print 4N bytes at ADDR of the memory. Show
in little endian
> b 0x80000f4
[WARN] (src/monitor.c:211: errno: None) Adding 0x00000000080000f4 breakpoint
succeeded.
> c
[INFO] (src/monitor.c:234) Hit breakpoints 0x00000000080000f4.
> info r
PC  : 0x00000000080000f4
x0  : 0x0000000000000000        x1  : 0x00000000080000c0
x2  : 0x0000000008008fc0        x3  : 0x0000000000000000
x4  : 0x0000000000000000        x5  : 0x0000000000000000
x6  : 0x0000000000000000        x7  : 0x0000000000000000
x8  : 0x0000000008008fe0        x9  : 0x0000000000000000
x10 : 0x0000000000000001        x11 : 0x0000000000000000
x12 : 0x0000000000000000        x13 : 0xffffffffffffffff
x14 : 0x0000000008000318        x15 : 0x0000000000000000
x16 : 0x0000000000000000        x17 : 0x0000000000000000
x18 : 0x0000000000000000        x19 : 0x0000000000000000
x20 : 0x0000000000000000        x21 : 0x0000000000000000
x22 : 0x0000000000000000        x23 : 0x0000000000000000
x24 : 0x0000000000000000        x25 : 0x0000000000000000
x26 : 0x0000000000000000        x27 : 0x0000000000000000
x28 : 0x0000000000000000        x29 : 0x0000000000000000
x30 : 0x0000000000000000        x31 : 0x0000000000000000
>
```

支持全部RV64IM指令集

支持系统调用(puts)

处理方式是转发rv64中的ecall指令到x86的syscall指令, 使用`unistd.h`作为外部系统调用接口。不直接使用syscall的考虑是:

1. 直接插入syscall不够方便，同时语义不明确
2. 插入x86 syscall汇编会降低跨平台的可移植性
3. 自己包装x86 syscall会增加代码复杂度, 并且做的事和标准C库同样的事情意义不大

运行:

```
make T=puts
```

输出:

```
-------Build Simulator-------
make[1]: Entering directory '/workspaces/Workspace/lab2_1/simulator/sim'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/workspaces/Workspace/lab2_1/simulator/sim'
-------Build Test-------
make[1]: Entering directory '/workspaces/Workspace/lab2_1/simulator/test'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/workspaces/Workspace/lab2_1/simulator/test'
-------Start Simulation-------
[INFO] (src/memory.c:74) Physical Memory Range:[0000000008000000,
000000000fffffff].
[INFO] (src/memory.c:80) The image is test/build/puts.bin, size = 17360.
Hello, World!
HIT GOOD TRAP!
[INFO] (src/cpu.c:33) Program ended at pc 080000a0, with exit code 0.
```

实现了memtrace功能

运行:

```
make T=dummy MEM_TRACE=ON
cat memtrace.out
```

输出:

```
-------Start Simulation-------
[INFO] (src/memory.c:98) Physical Memory Range:[0000000008000000,
000000000fffffff].
[INFO] (src/memory.c:104) The image is test/build/dummy.bin, size = 176.
HIT GOOD TRAP!
[INFO] (src/cpu.c:33) Program ended at pc 0800004c, with exit code 0.
make: use "cat memtrace.out" to check the memtrace.
root@de5656b9e044:/workspaces/Workspace/lab2_1/simulator# cat memtrace.out
w 0x0000000008008ff8 8 0000000008000010
w 0x0000000008008ff0 8 0000000000000000
w 0x0000000008008fd8 8 0000000008009000
r 0x0000000008008fd8 8 0000000008009000
w 0x0000000008008fec 4 00000000
```

```
r 0x0000000008008fec 4 00000000
w 0x0000000008008fd8 8 0000000008009000
w 0x0000000008008fcc 4 00000000
r 0x0000000008008fcc 4 00000000
```

Remark

由于链接libc puts对内存布局的要求，将MEM_BASE设置为了0x08000000