Spike 模拟器snapshot功能技术方案及实现分享

王萌 from PLCT实验室 2020.07.05

目录

Spike简介

- 一个开源的基于RISC-V指令集架构的模拟器，使用C++编写
- 实现了一个或多个risc-v harts的功能模拟
- 由sifive进行维护
- 支持RV32/64，并且提供最新版本指令集扩展的支持
- repo：https://github.com/riscv/riscv-isa-sim

Spike简介

- spike提供了一套不基于特定硬件实现的纯软件模拟
- 可模拟指令级别的行为，包括寄存器和内存操作
- 可以用于指令行为的测试
- 同时提供调试功能，提供了交互式的debug interface
- spike也可以被模拟成一个真实的设备，通过openocd&gdb进行调试

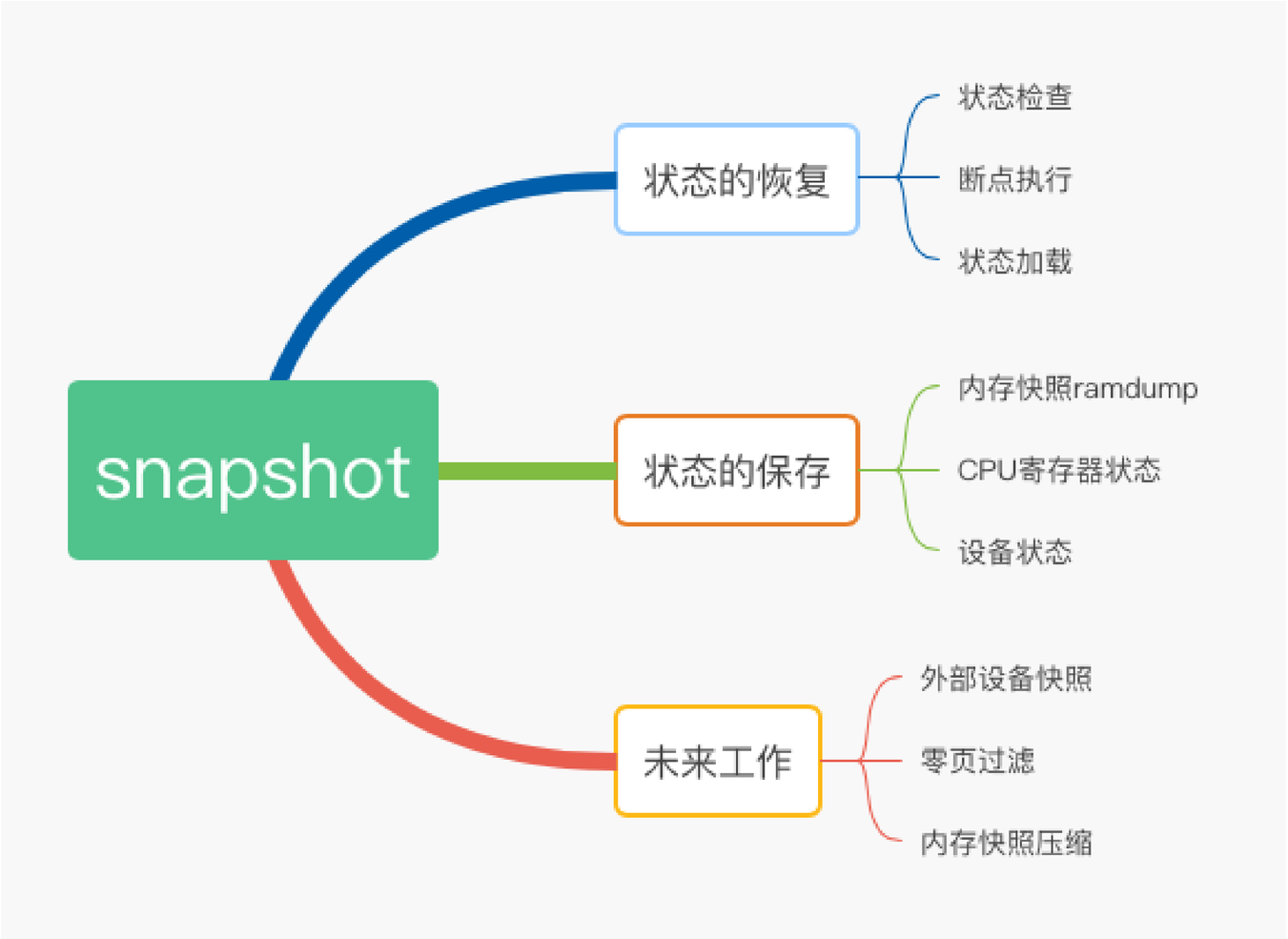
snapshot功能概述

- snapshot(快照)功能，可以用来保存系统运行到某一时刻的状态，提供某一时刻系统运行状态的完全拷贝
- 可以提供系统的checkpoint，系统可以随时恢复到某个还原点继续运行
- 我们将在spike上实现快照功能，包括快照的生成与加载
- 可以将程序的运行状态保存为快照
- 可以在不同设备上加载相同快照实现状态还原

snapshot功能的作用

- 当模拟的程序运行错误时，可以通过快照保存出错时的状态，供其他开发者分析（崩溃调试）
- 可以将程序中某些特定的不容易到达的状态保存为快照，方便多次进行调试，减少了重复操作

snapshot整体方案



snapshot功能实现

- 内存部分：
  - 需要实现内存的dump操作
  - 这一部分已经完成了对应的代码实现
  - 后续会考虑使用零页过滤，压缩等手段减小内存文件体积

```
bool sim_t::ramdump(const char * path)
{
    ofstream out;
    char c = '$';
    out.open(path,ios::in|ios::out|ios::binary);
    for(auto &mem:mems)
    {
        out.write((char*)&mem.first,sizeof(mem.first)); //base address
        size_t size = (*mem.second).size();
        out.write((char*)&size,sizeof(size_t));
        out.write((char *)mem.second->contents(),size);
        out.write((char* )&c,sizeof(char));
    }
    return true;
}
```

snapshot功能实现

- CPU部分：
  - 需要保存CPU运行过程中产生的所有状态
  - spike的实现中为每个hart提供了一个state_t用于保存状态
  - 需要进一步测试以确定是否包含了全部状态

```
// architectural state of a RISC-V hart
struct state_t
{
  void reset(reg_t max_isa);

  static const int num_triggers = 4;

  reg_t pc;
  regfile_t<reg_t, NXPR, true> XPR;
  regfile_t<freg_t, NFPR, false> FPR;

  // control and status registers
  reg_t prv;
  reg_t misa;
  reg_t mstatus;
  //...
```

snapshot功能实现

- 设备部分：
  - spike中目前只支持有限的设备，包括总线设备，内存设备，用于控制定时器中断和软件中断的中断控制器设备，以及一个用于实现自定义MMIO设备的mmio_plugin
  - ——更正，中断控制器clint也是需要保存的
  - 这些设备暂时不需要模拟，在spike初始化时会自动初始化设备
  - 关于spike中设备模拟的更多信息可参考PLCT实验室往期报告
  - https://www.bilibili.com/video/BV1ET4y1J7As

snapshot功能实现

- 其他:
  - 在spike中提供了一套用于实现系统调用和I/O操作的interface
  - 在fesvr(RISC-V front-end server)模块中的htif对象中实现
  - 在这部分中也存储着一些程序运行必要的参数和状态等，这部分可以通过在恢复状态时加载ELF文件实现

实现进度及目标

- 目前，内存部分的实现已经完成，可以保存dump文件，加载dump并继续运行程序
- 接下来的工作：
  - 实现CPU状态的保存
  - 快照的打包
  - 通过零页过滤，压缩等方式减小内存文件体积

```
[root@iZ2zeccgxm1p6kwvtxkqvpZ riscv-isa-sim]# spike -m512 -d pk hello
: ramdump test
ramdump file saved to test
:
```

```
[root@iZ2zeccgxm1p6kwvtxkqvpZ riscv-isa-sim]# spike --ramdump test pk hello
bbl loader
hello,world!
[root@iZ2zeccgxm1p6kwvtxkqvpZ riscv-isa-sim]#
```

未来工作

- 完成完整的snapshot功能，并将成果通过upstream PR等方式进行开源
- 我们的fork仓库地址：https://github.com/isrc-cas/plct-spike
- 后续我们还会在spike上继续进行开发
- 我们还计划为spike提供新的I/O设备，使spike模拟的target能够与主机host进行通信

讨论

- 我们也希望听取大家的意见和建议：
  - 报告内容中存在的问题
  - 对我们目前实现的建议和看法
  - 希望spike添加的feature
  - 畅所欲言

Thanks for listening!