香山分支预测器内外快速性能评估工具
Branch Predictor Related Performance Evaluation Tools of XiangShan

陈国凯 周耀阳
中科院计算所
2022年8月

实现工作

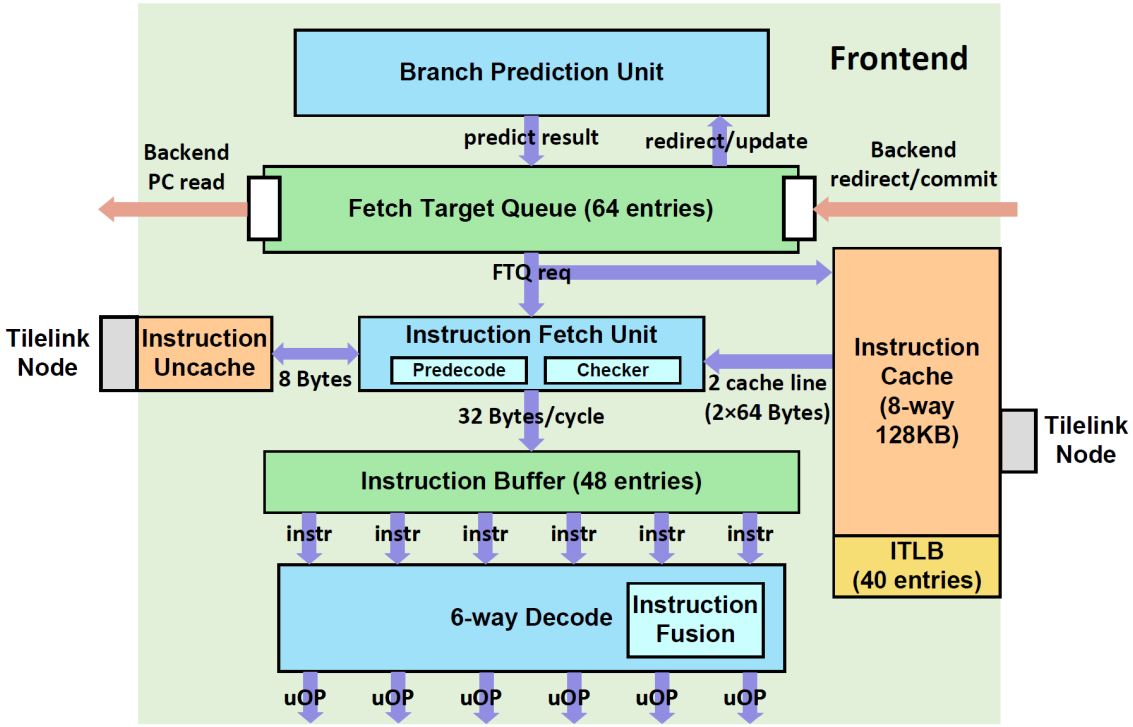
- Oracle Branch Predictor
  - 性能评估的 roofline 模型
- Branch Predictor Unit Tester
  - 拆分预测器独立仿真测试

Oracle Branch Predictor

- 问题：高性能处理器性能探索时各部件间有紧密关联
  - 当前较低的分支预测器准确率可能掩盖其他功能部件的性能提升点
- 解决：利用外部正确分支信息辅助生成预测结果的预测部件
- 实现层次：RTL
  - 软件模拟器（如 Gem5）理论上可实现准确评估，但对齐工作量大且需不断同步 RTL 进展
  - 当前 RTL 中分支预测器接口较为简洁，维护工作量可控

Decoupled Frontend

- 香山处理器前端取指架构的演进
- 分支预测器相关概述
  - 分支预测以预测块为单位
  - 每个预测块记录至多2条分支/跳转指令的 PC、类型
  - 预测块在 FTQ 生成，随后以 FTB 项形式缓存在 BPU 内 FTB 模块中
  - 预测器对块内分支指令进行预测
  - 含冲刷多级预测器

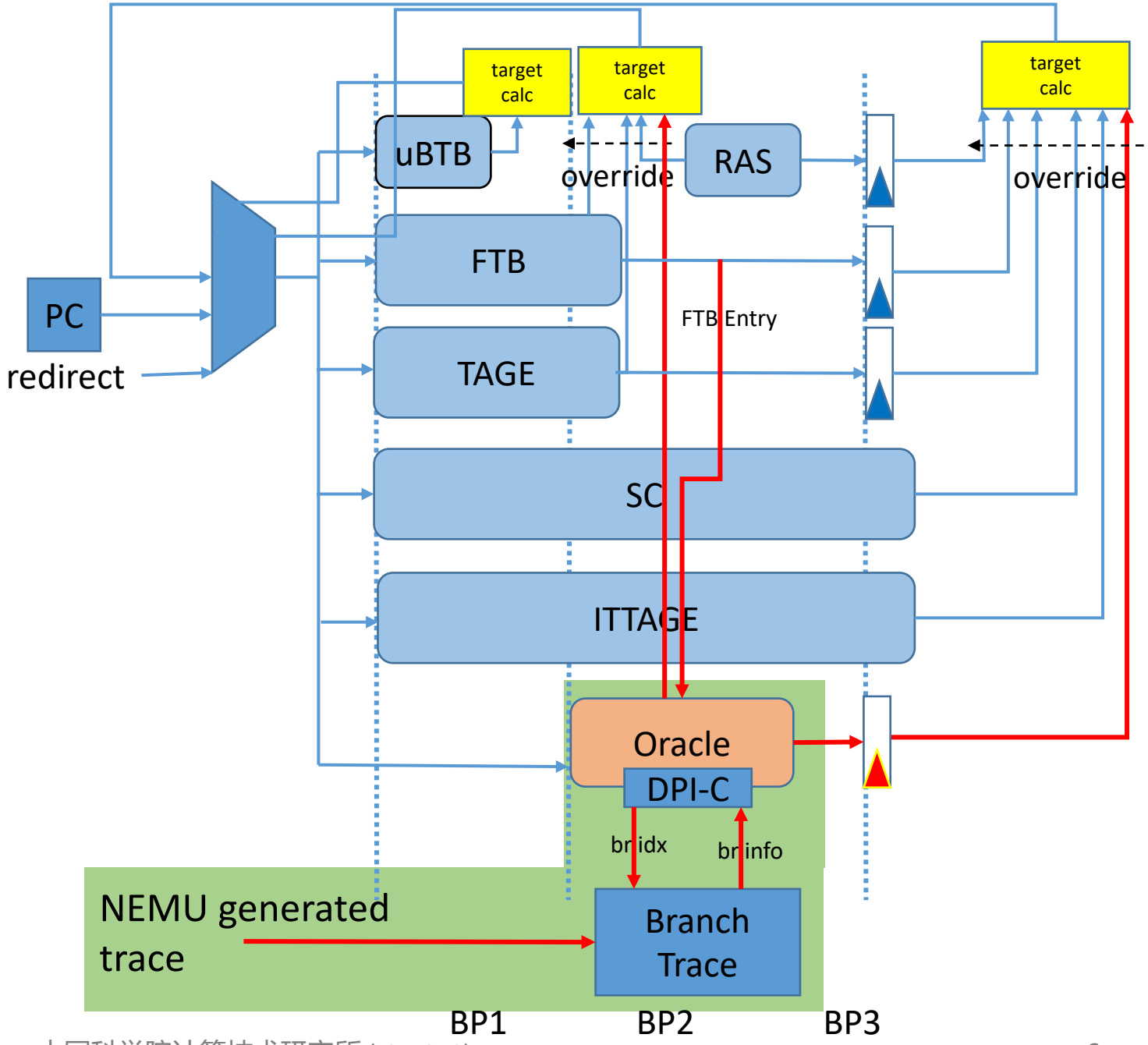


功能预期

- 需要实现
  - 能够在 FTB 项正确情况下为其生成正确的预测结果
  - 正确结果包含指令的跳转方向和地址
  - FTB 项未命中/命中项不正确时不影响后续正常工作
- 不关注
  - FTB 项未命中/命中项不正确/其他与实际不一致情况的 FTB 项数据纠正

实现细节

- NEMU 模拟器（@2021 峰会）导出正确分支路径
- 正确路径动态传递给 Oracle Predictor
- Oracle Predictor 根据分支指令信息查找匹配指令，若匹配则覆盖之前预测器预测结果



实现细节

- 分支预测准确率控制
  - 预处理分支预测结果，随机反转方向
  - 被反转记录会触发分支方向预测错误

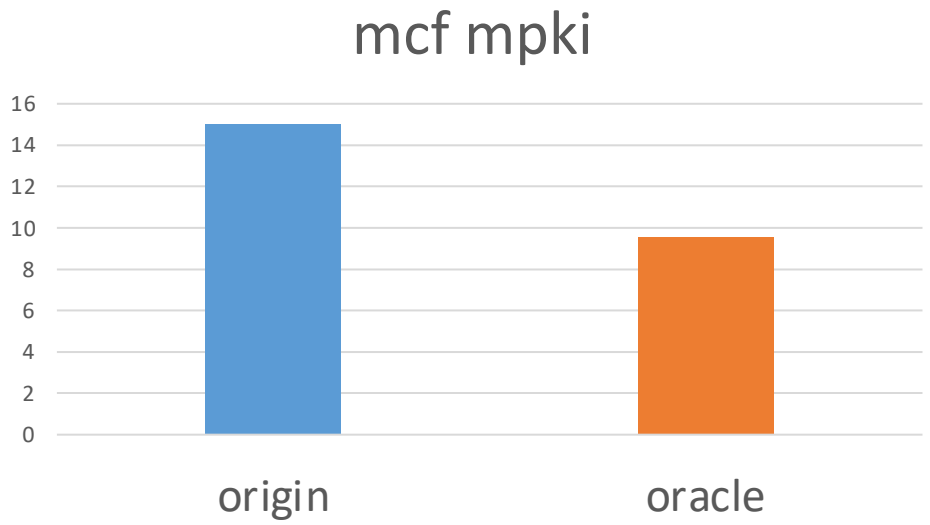
```
if (rand() % 100 < miss_rate) {
  record[i].taken = ~record[i].taken;
  num_reverted++;
}
```

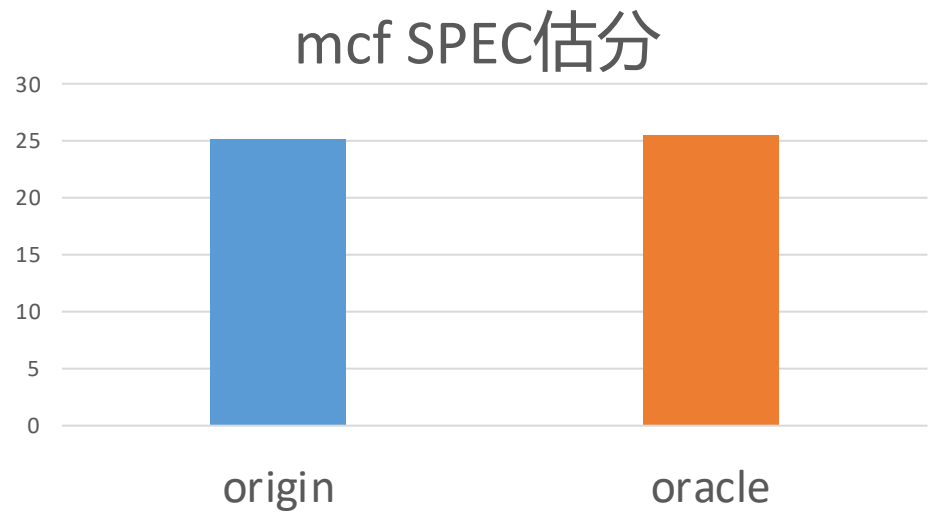
实现细节

- 误预测维护
  - 必要性
    - Oracle 预测器预测准确率可调整，可能生成错误结果
    - 即使 Oracle 预测器被提供 100% 正确信息，仍可能因 FTB 项出错产生误预测
  - 维护信息
    - 当前执行位置在正确路径上的分支指令编号作为 meta 信息传递给后续流水级与分支预测结果一同存储下来
    - 后续误预测的重定向请求携带该编号信息用于恢复正确路径分支指令编号

案例数据

- 证明 SPEC CPU mcf 测试瓶颈与分支预测关联较小





缺陷与改进方向

- 分支预测准确率无法精确反映误预测指令分布与流水线指令供应情况
  - 预测器预测错误的指令**并非随机分布**
  - 发生在不同时刻的不同种类分支预测错误有不同的流水线清空代价
    - Predecode、ALU、Jump Unit
    - 需要更为细节的微结构信息作为参考
      - ROB 中被刷掉的指令
      - ROB 中因误预测被刷掉的执行完毕待提交指令
      - LSU 中因误预测将被刷掉的已发出但尚未收到回复访存指令
    - 数据导出
      - 仿真时在 RTL 对应模块导出
- WIP：采用深度学习等手段模拟实现更精确的分支预测错误模拟

运行步骤

- 所涉源码位置
  - NEMU
    - Trace 生成
      - https://github.com/OpenXiangShan/NEMU/tree/oracle_trace
    - Difftest
      - 使用香山 repo 中 ready_to_run 即可
  - 香山
    - https://github.com/OpenXiangShan/XiangShan/tree/decoupled-oracle
  - Difftest
    - https://github.com/OpenXiangShan/difftest/tree/oracle

运行步骤

- 编译可导出 trace 版本NEMU
  - make riscv64-xs-btrace_defconfig
- 运行 workload（以checkpoint为例）
  - ./build/riscv64-nemu-interpreter -b -I 40000000 -D ./output -w CHECKPOINT_NAME -C CHECKPOINT_TIME -c /CHECKPOINT_PATH/CHECKPOINT.gz
- 编译香山
- 运行同样 workload，指明 trace 文件 PATH 及可选 miss rate
  - . /build/emu -e 0 -i CHECKPOINT_PATH/CHECKPOINT.gz -I 100000 -r BRANCH_TRACE_PATH --miss-rate 10

运行效果

```
total guest instructions = 100,000
instrCnt = 100,000, cycleCnt = 97,815, IPC = 1.022338
Seed=0 Guest cycle spent: 97,817 (this will be differe
Host time spent: 14,505ms
```

开启 Oracle Branch Predictor

```
total guest instructions = 100,000
instrCnt = 100,000, cycleCnt = 108,721, IPC = 0.919786
Seed=0 Guest cycle spent: 108,723 (this will be differ
Host time spent: 17,481ms
```

关闭 Oracle Branch Predictor

Branch Predictor Unit Tester

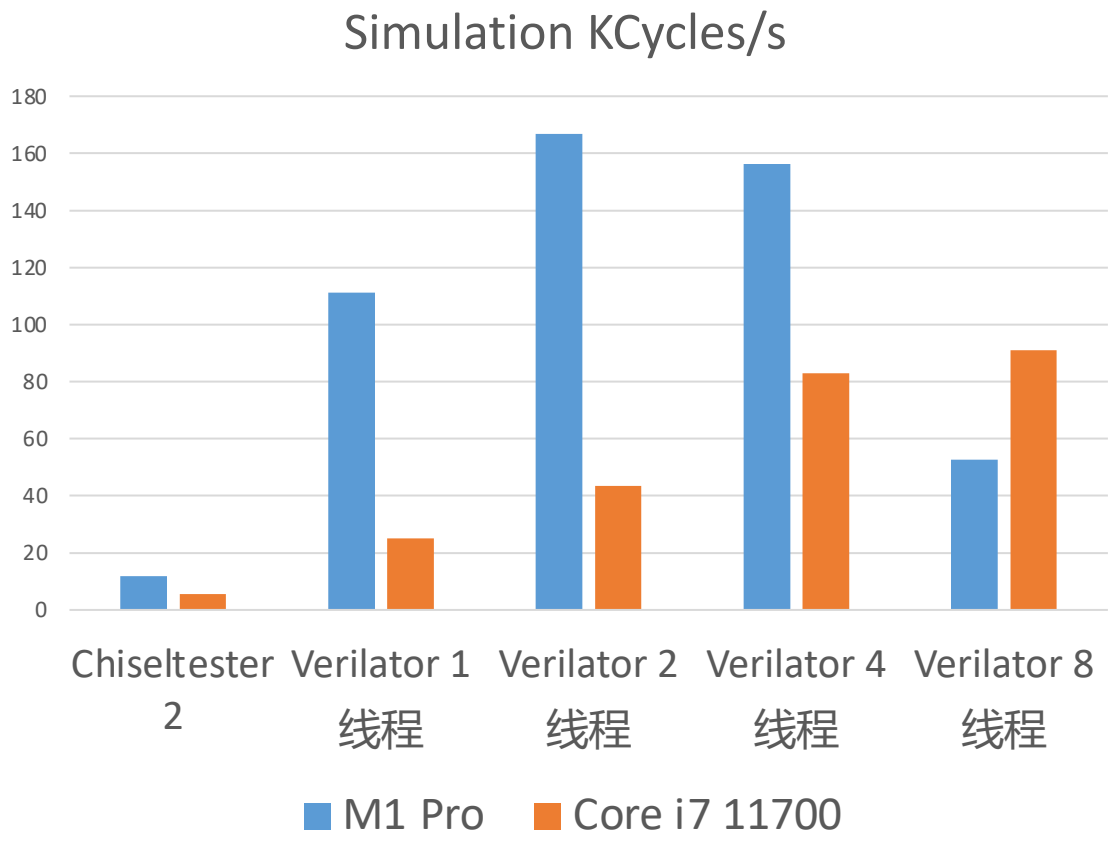
- 分支预测器是性能探索重点
- 在模拟器进行性能探索
  - RTL 与模拟器对齐难度大，对齐后仍需在 RTL 上完成最终调试
- 通过处理器完整 RTL 仿真探索
  - 规模巨大，仿真效率较低 ~ 10K指令/s
- 剥离分支预测器 RTL，搭建单元级交互模型
  - 分支预测单元规模可控：只需模拟部分接口，还可复用香山框架内既有性能计数器、日志等机制

实现目标

- 独立的分支预测器测试框架
  - 模拟分支预测器外部接口，以尽可能高准确度快速仿真分支预测器，评估其实现效果

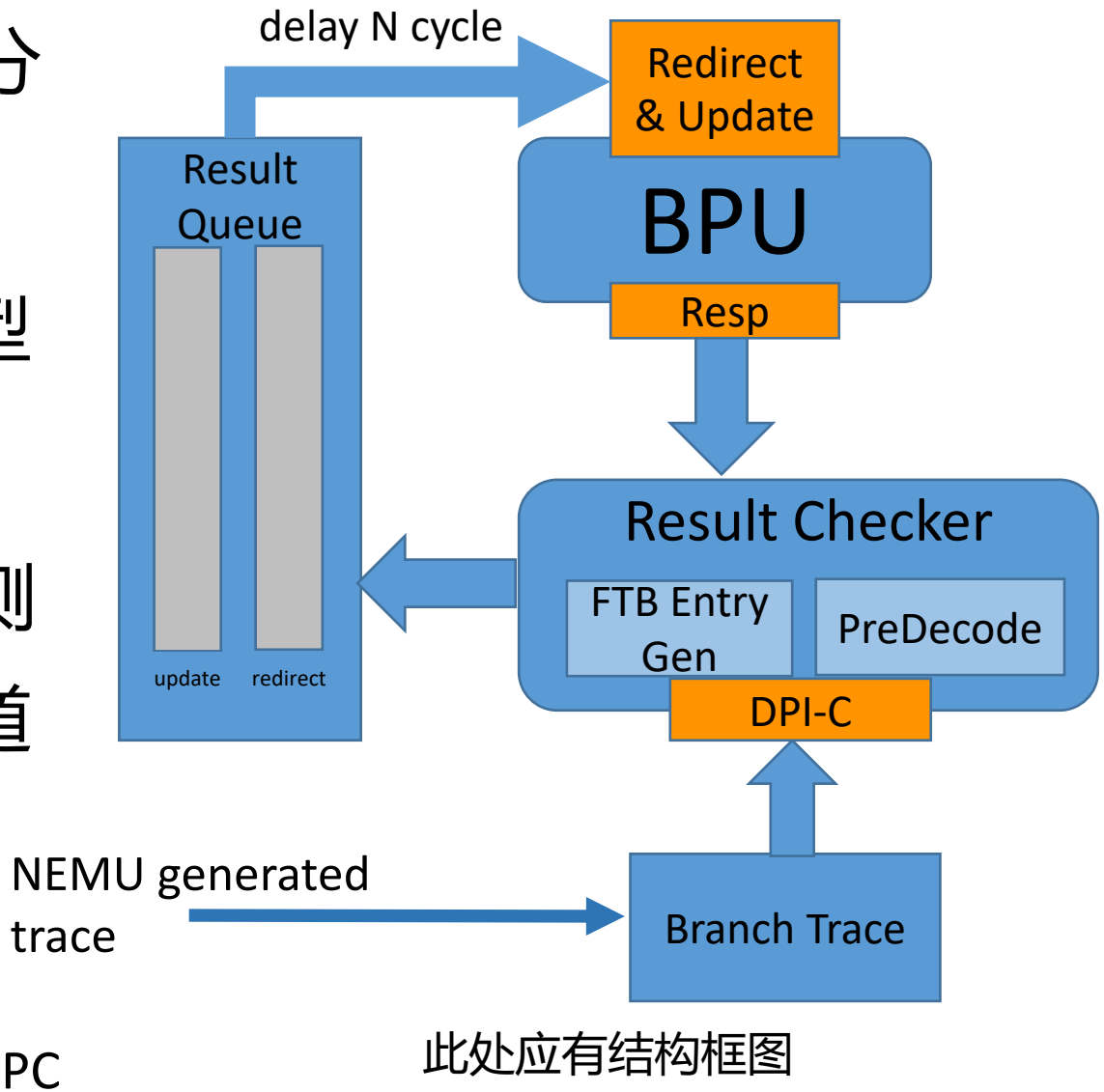
已有工具的探索选择

- ChiselTest：ucb-var/chisel-testers2
  - 为 Chisel 开发的 RTL 设计的测试工具
  - 使用 scala，便于与 Chisel RTL 结合
  - 效率较低，不适于本场景
- Verilator
  - Verilog 仿真器，效率较高
  - 支持编写测试激励，但无法识别 Chisel 高级结构



实现细节

- 利用 NEMU 模拟器导出执行路径中分支指令
- 利用 DPI-C 接口将分支指令 PC、类型等信息传递给测试 Wrapper 模块
- 测试 Wrapper 模块内实例化分支预测单元，利用 update 和 redirect 两通道与分支预测单元交互
  - update 通道负责更新 FTB
  - redirect 通道负责 PC 地址初始化及误预测 PC 纠正



此处应有结构框图

实现细节

- 预测结果 update、 redirect 延迟
  - 真实处理器中 update、 redirect 时机与指令执行时间关联
  - 需要支持延迟的动态调整
    - update、 redirect 信息在 s3 结果输出当拍产生，与延迟信息一同传递给 CPP 部分队列暂存
    - CPP 部分根据延迟信息择机发送请求
- verilator 技术细节
  - verilator 不掌握 Chisel 的 Bundle 封装高层次信息
    - 逐信号连接将需要维护大量信号
  - Chisel 内打包为一个信号再与 verilator 交互

当前不足与改进

- FTB Entry 生成逻辑与真实 RTL 行为一致性检查困难
  - 计划采用可变延迟功能将延迟与真实场景周期对齐，而后精确比对 FTB Entry 内数据

敬请批评指正