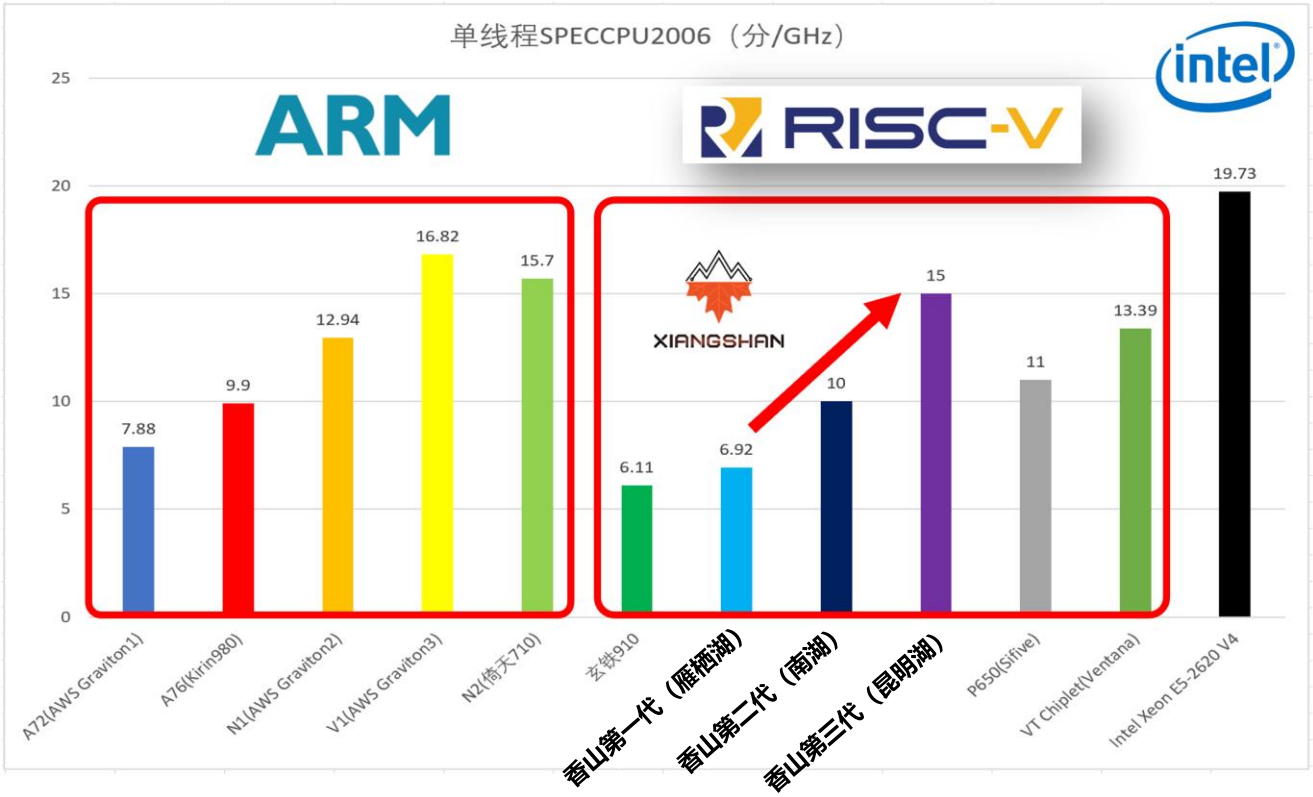
敏捷开发工具链助力香山高性能处理器核

王凯帆 徐易难 陈国凯 李昕 等

中国科学院计算技术研究所

2023 年 8 月 24 日@第三届 RISC-V 中国峰会

背景：高性能处理器竞争激烈

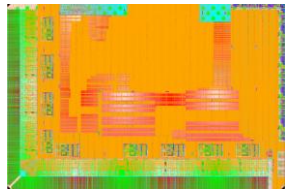


- RISC-V 处理器核与 **X86/ARM** 目前在性能上**存在差距**
- RISC-V 阵营内部，国际企业 **SiFive** 和 **Ventana** 等具备一定的先发优势和人才优势

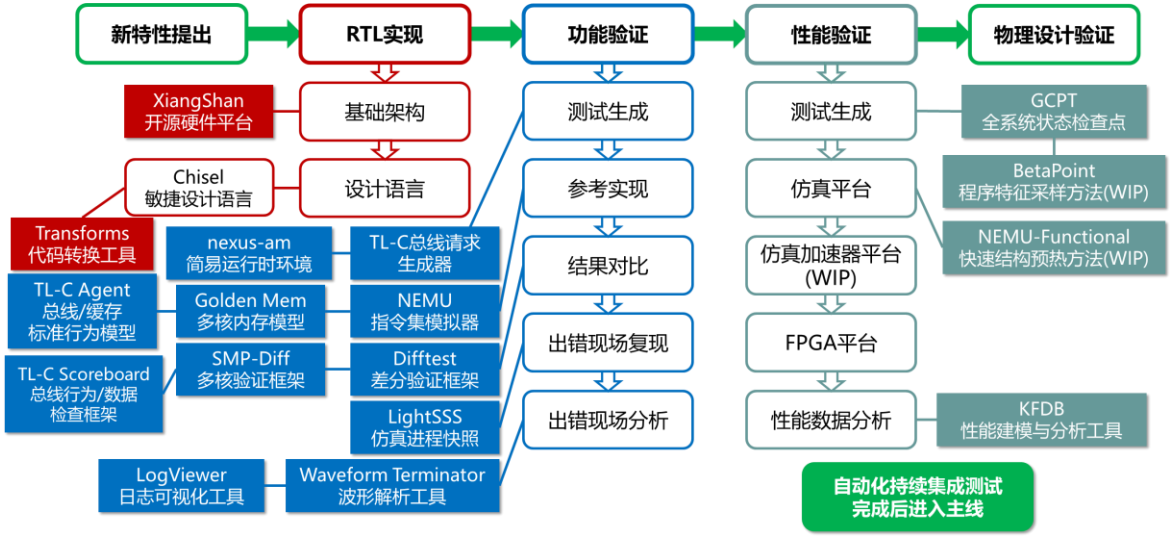
➢ **香山处理器核需要通过敏捷开发工具链提升迭代速度，以快速实现对标甚至超越**

背景：高性能处理器敏捷开发

- 香山核心价值是构建一套**芯片敏捷设计基础设施**，缩短迭代优化周期
- **开源开放能力体系**，联合企业加速处理器研发节奏，支持按需快速定制芯片



成果开源



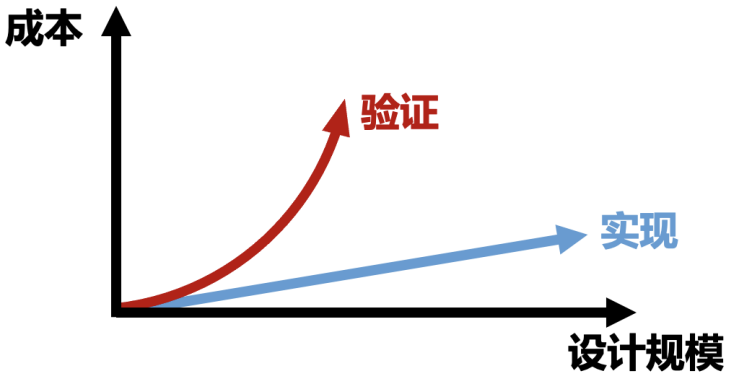
能力开放

高性能处理器敏捷开发现状

- 香山高性能处理器核开发进入**深水区**，面临的两个主要的问题

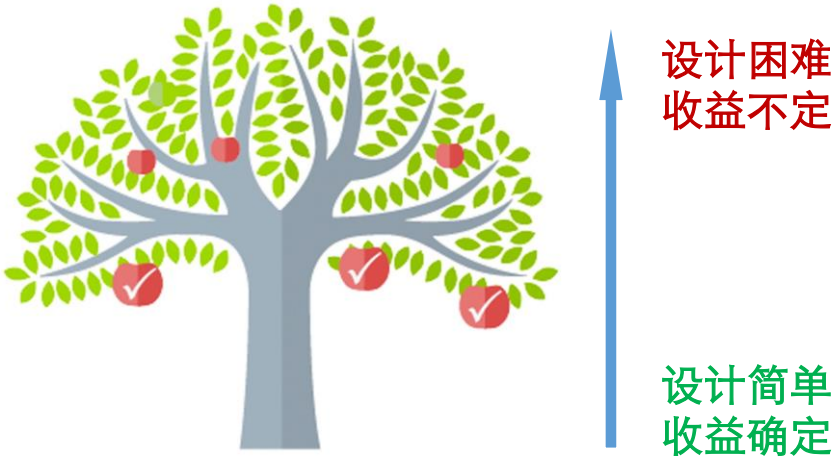
功能验证

处理器设计与验证在敏捷度上的差距持续扩大，形成一堵验证墙



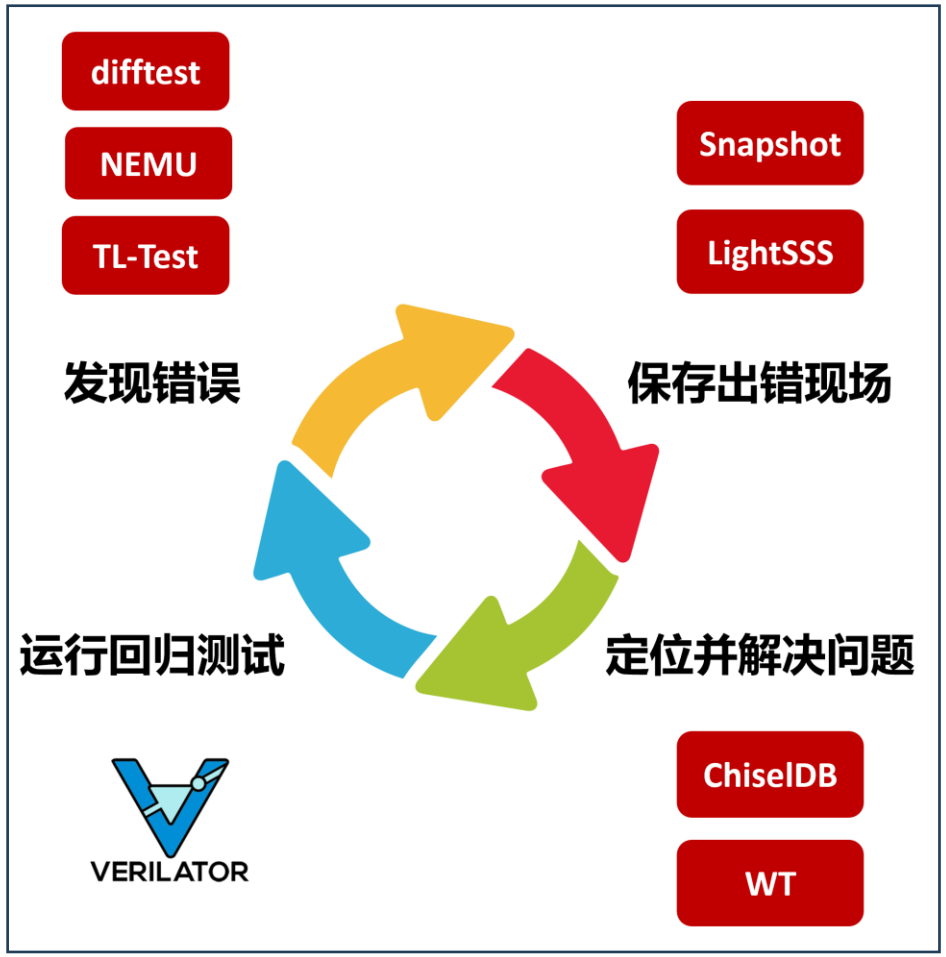
性能优化

已知简单优化点实现殆尽，性能优化转向探索性与精细化

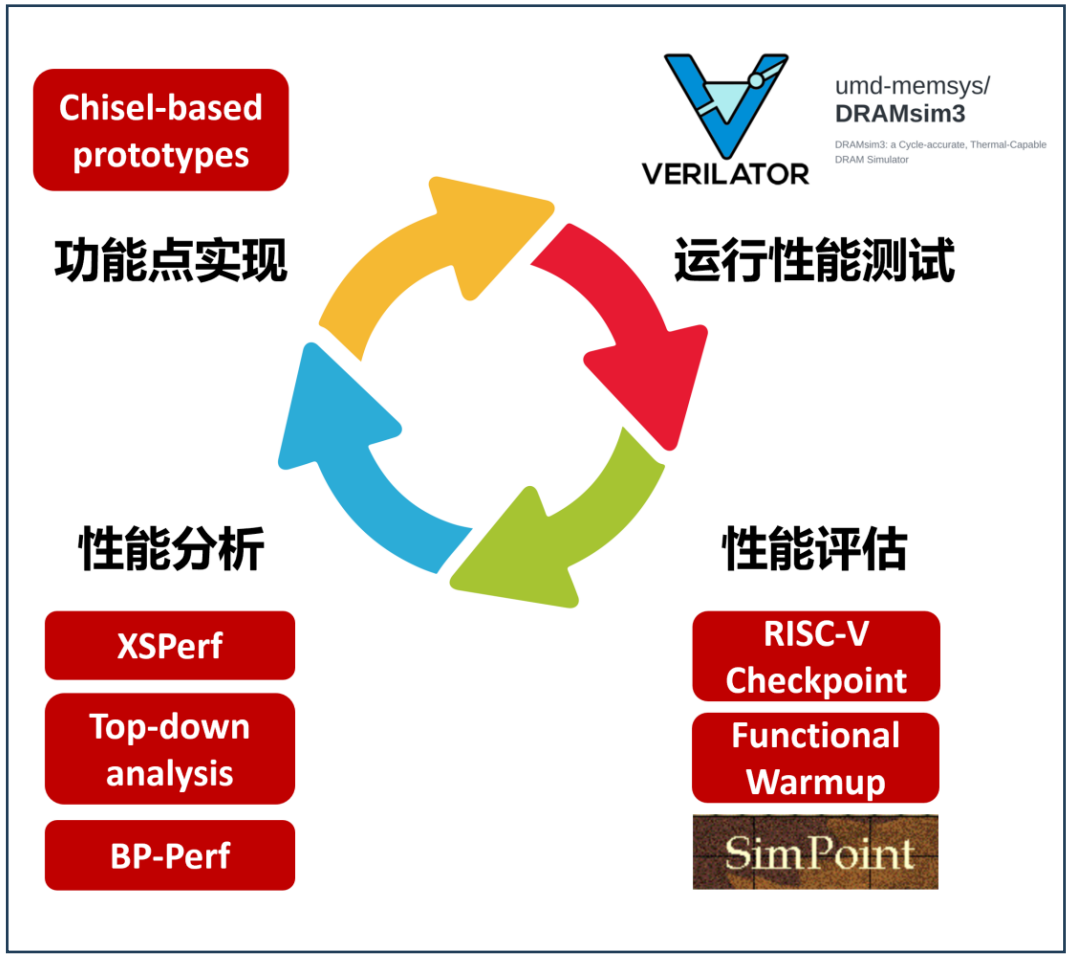


回顾：香山敏捷开发工具链

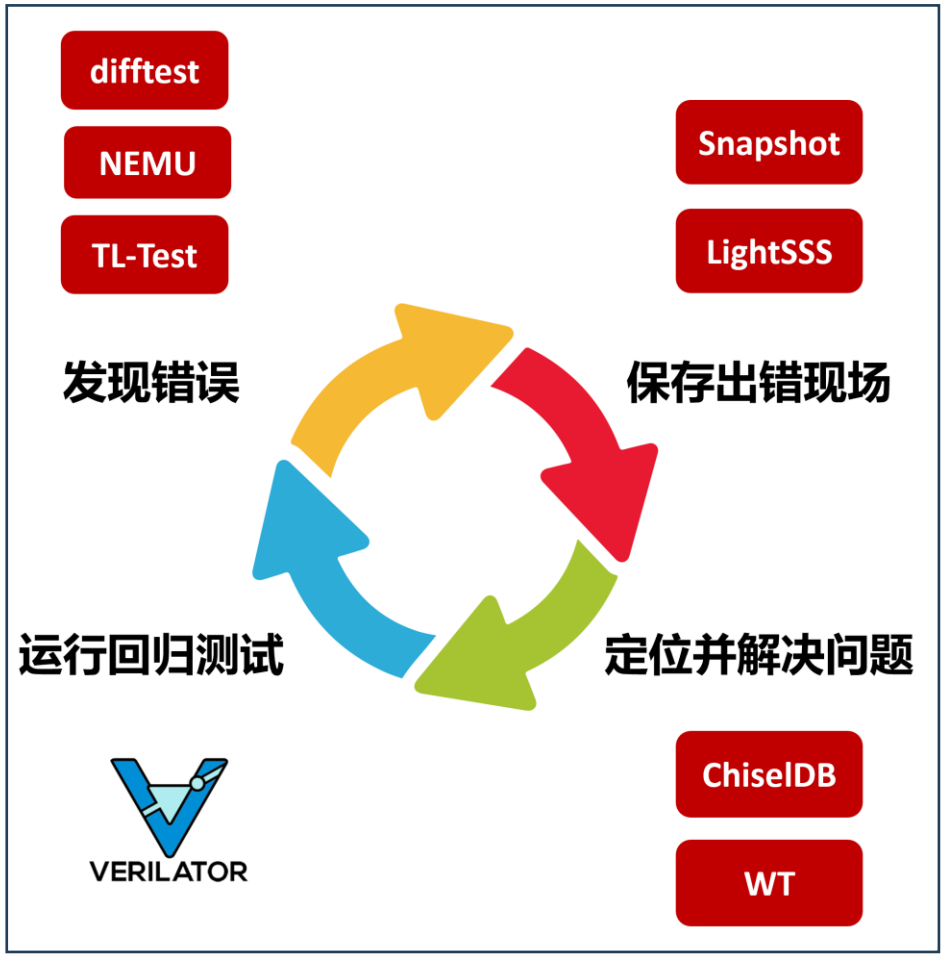
- ❶ 敏捷功能验证



- ❷ 敏捷性能优化



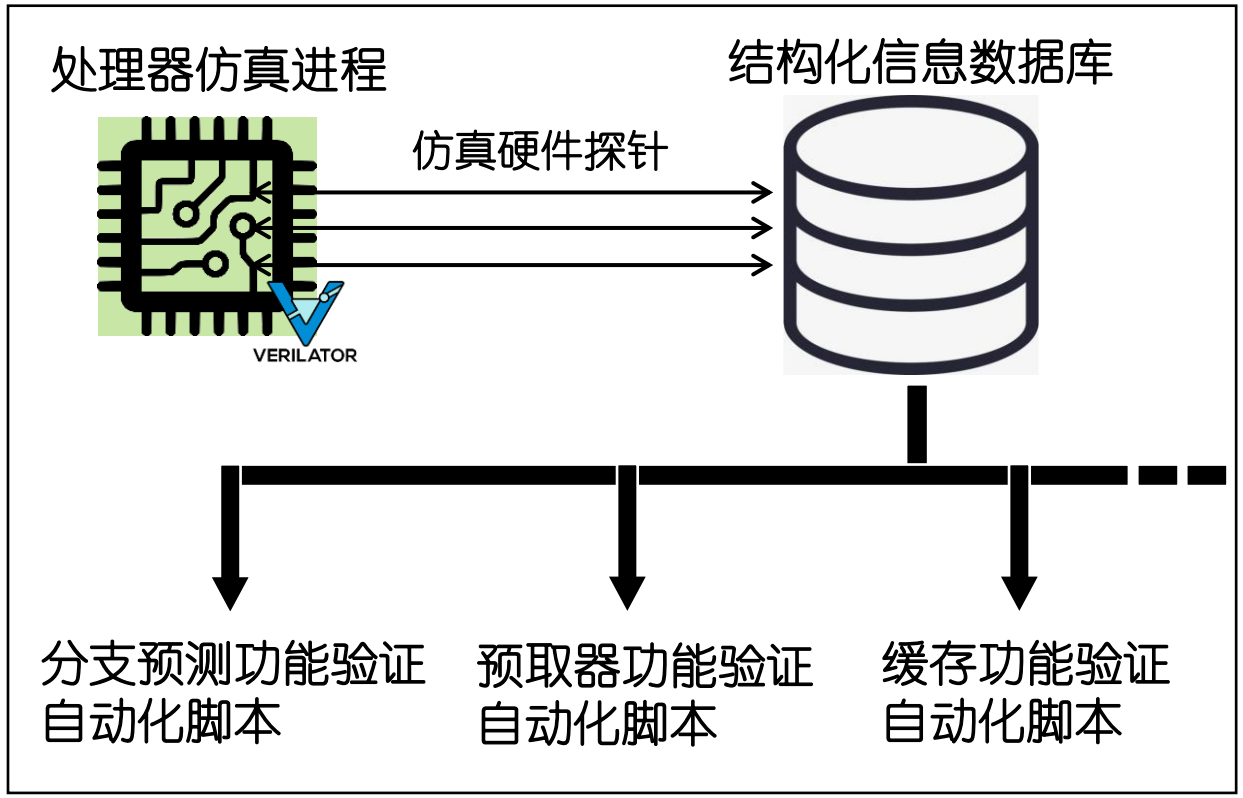
功能验证循环的不足

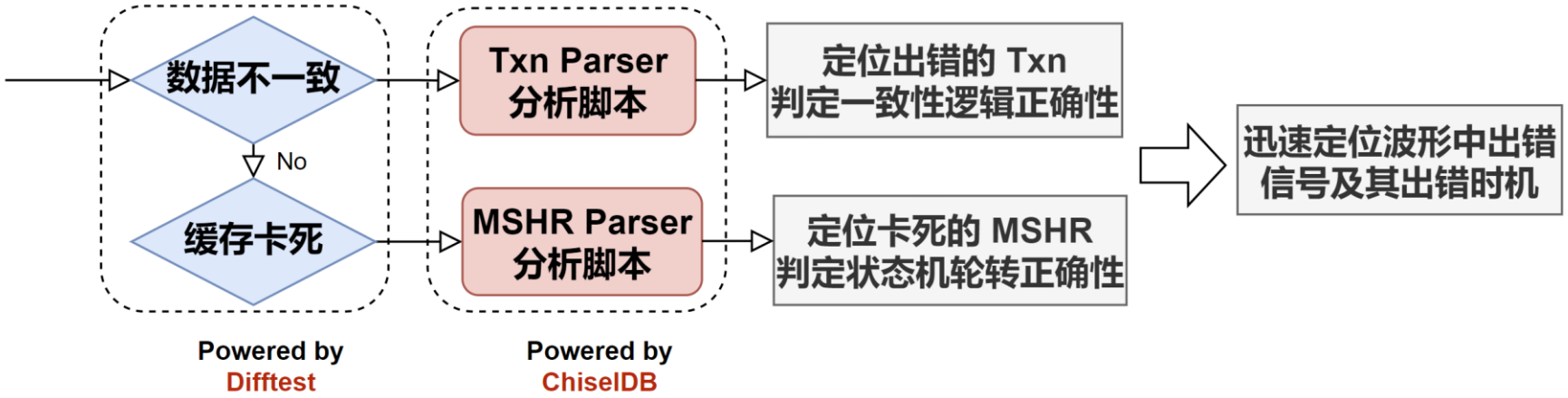


- **问题定位环节仍然需要大量人力**
  - ➢ 结构化信息辅助调试
- **系统级验证提升覆盖率困难**
  - ➢ 硬件模糊测试

结构化信息辅助调试

- 功能验证的**最后一公里**问题
  - **如何根据硬件出错状态定位 Bug**
- ChiselDB[1] 为**结构化信息调试**提供可能
- 以处理器缓存调试为例：

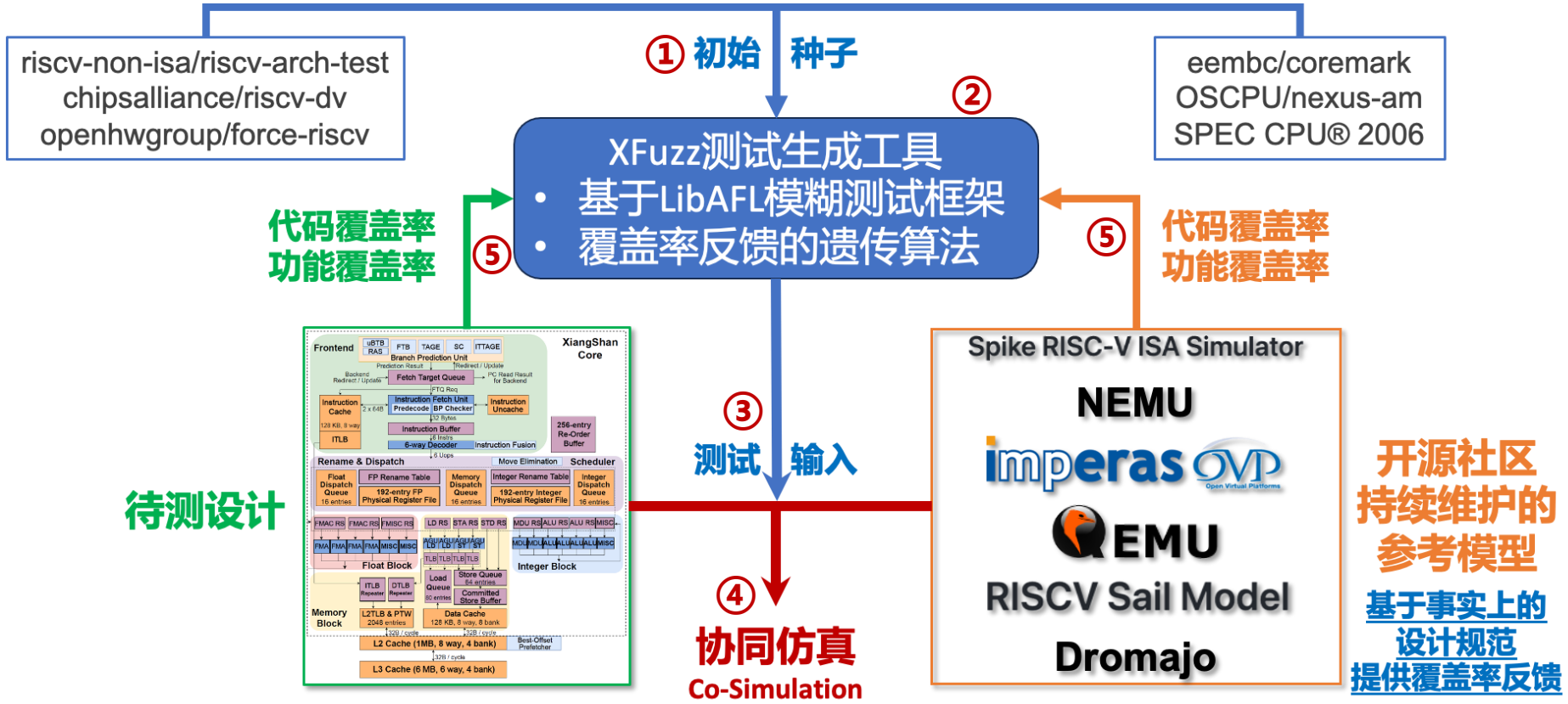




[1] ChiselDB: Mapping Hardware Data Structures to Database for Efficient Data Analysis, Jiawei Lin@ICT

面向 RISC-V CPU 的覆盖率导向模糊测试

- 原理：①初始种子 → ②遗传算法 → ③新测试输入 → ④错误检查
  ← ⑤覆盖率反馈



面向 RISC-V CPU 的覆盖率导向模糊测试

- 应用：指导发现大量功能正确性 BUG

```
[02] commit pc 00000000800000a8 inst 00000f93 wen 1 dst 31 data 0000000080000004 li     t6, 0
[03] commit pc 00000000800000ac inst f1402573 wen 1 dst 10 data 0000000000000000 csrr   a0, mhartid
[04] commit pc 00000000800000b0 inst 00051063 wen 0 dst 00 data 0000000080000004 bnez   a0, pc + 0
[05] commit pc 00000000800000b4 inst 00000297 wen 1 dst 05 data 00000000800000b4 auipc  t0, 0x0
[06] commit pc 00000000800000b8 inst 01028293 wen 1 dst 05 data 00000000800000c4 addi   t0, t0, 16
[07] commit pc 00000000800000bc inst 30529073 wen 0 dst 00 data 0000000000000000 csrw   mtvec, t0
[08] commit pc 00000000800000c0 inst 18005073 wen 0 dst 00 data 0000000000000000 csrwi  satp, 0
[09] commit pc 00000000800000c4 inst 00000297 wen 1 dst 05 data 00000000800000c4 auipc  t0, 0x0
[10] commit pc 00000000800000c8 inst 02828293 wen 1 dst 05 data 00000000800000ec addi   t0, t0, 40
[11] commit pc 00000000800000cc inst 30529073 wen 0 dst 00 data 00000000800000c4 csrw   mtvec, t0
[12] commit pc 00000000800000d0 inst 0010029b wen 1 dst 05 data 0000000000000001 addiw  t0, zero, 1
[13] commit pc 00000000800000d4 inst 03529293 wen 1 dst 05 data 0020000000000000 slli   t0, t0, 53
[14] commit pc 00000000800000d8 inst fff28293 wen 1 dst 05 data 001fffffffffffff addi   t0, t0, -1
[15] commit pc 00000000800000dc inst 3b129073 wen 0 dst 00 data 00000002535cac00 csrw   pmpaddr1, t0 <--
[16] commit pc 0000000080000060 inst 00000693 wen 1 dst 13 data 0000000080000004 li     a3, 0
[17] commit pc 0000000080000064 inst 00000713 wen 1 dst 14 data 0000000080000004 li     a4, 0
```

- access-fault for non-leaf out-of-range PTE access: poemonsense@ `ff025a6`

```
diff --git a/src/main/scala/rocket/PTW.scala b/src/main/scala/rocket/PTW.scala
index 7c36bdbf3..8701caa0b 100644
--- a/src/main/scala/rocket/PTW.scala
+++ b/src/main/scala/rocket/PTW.scala
@@ -720,6 +720,7 @@ class PTW(n: Int)(implicit edge: TLEdgeOut, p: Parameters) extends CoreModule()(
           resp_valid(r_req_dest) := true.B
         }

+        resp_ae_ptw := ae && count < (pgLevels-1).U && pte.table()
         resp_ae_final := ae
         resp_pf := pf && !stage2
         resp_gf := gf || (pf && stage2)
```

For non-leaf PTEs (`pte.table()` is set, and `count` is less than the `pgLevels`), we should set `resp_ae_ptw` instead of `resp_ae_final` . Besides, since `resp_ae_final` would have lower priority than page-fault and `ae_ptw` access-fault, we don't need to shrink its condition and keep the condition as simple as possible.

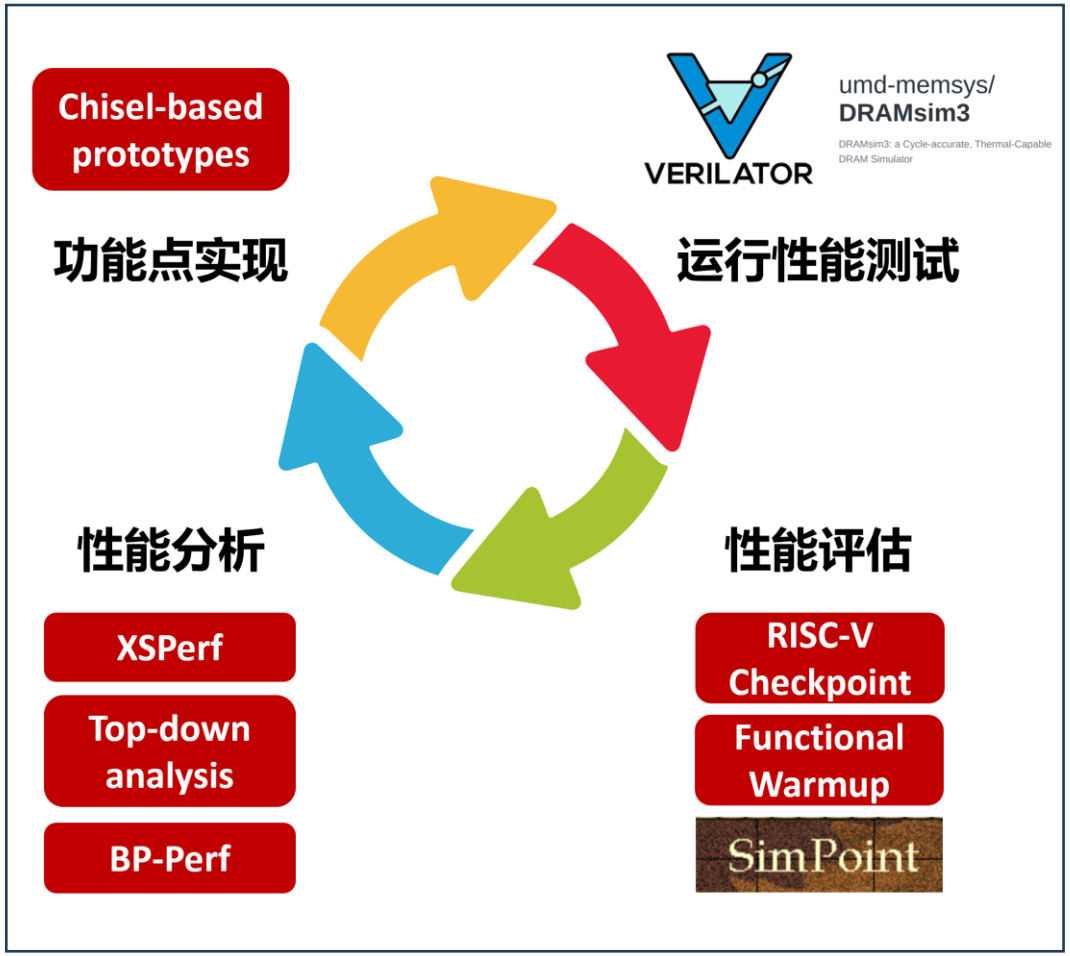
落地香山第二代南湖项目

错误示例：pmpaddr1 寄存器逻辑实现错误

测试其他开源处理器功能正确性

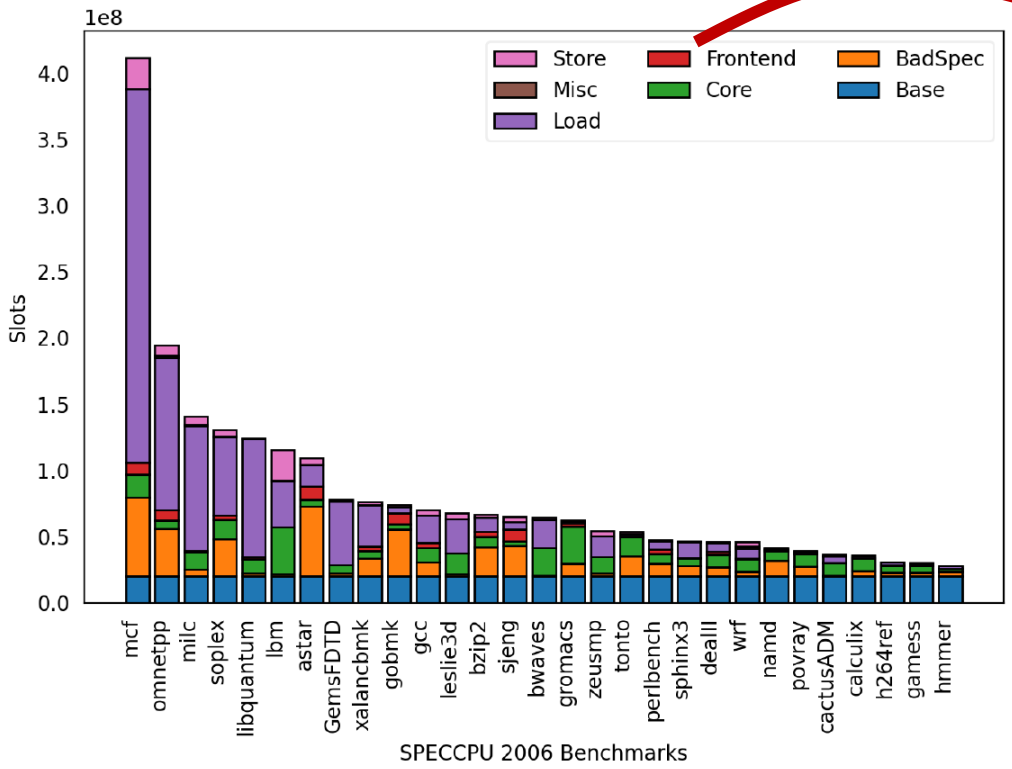
rocket-chip：越界访问页表时未正确报出 access-fault 异常

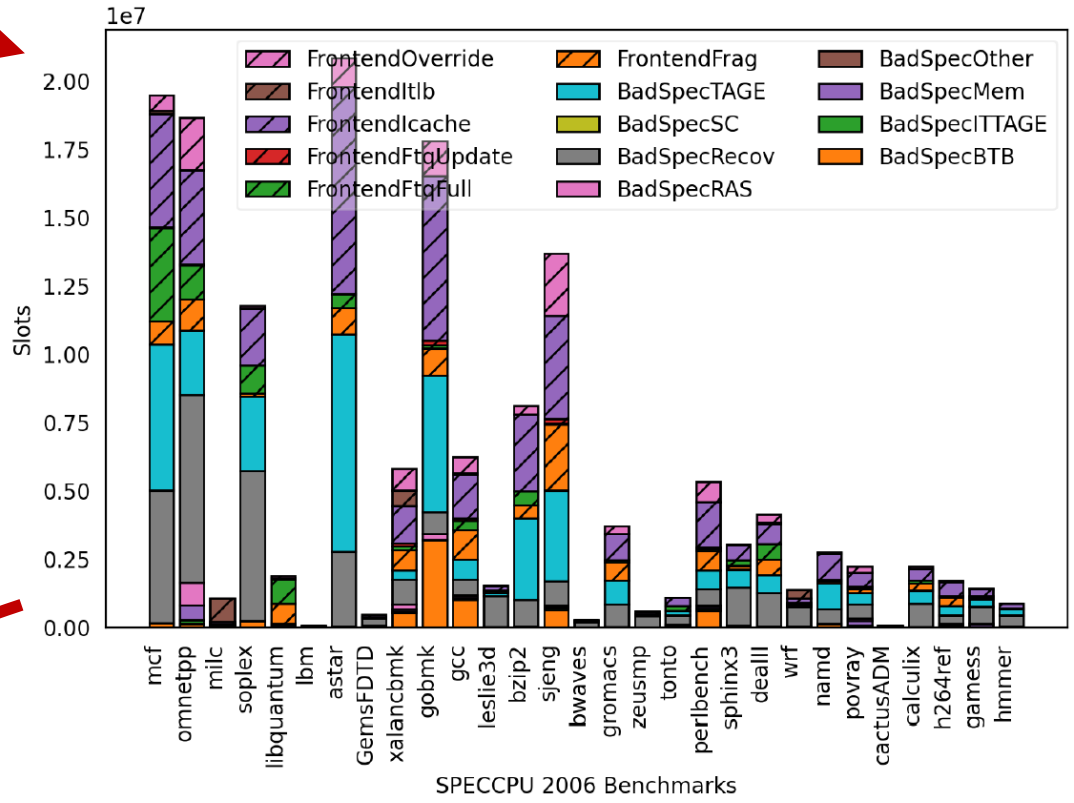
性能优化循环的不足



- **精细化**
  - ➢ 从宏观到微观的性能分析流程
- **探索性**
  - ➢ 运行时设计参数调整
  - ➢ 设计参数自动求解

Top-down 宏观性能分析

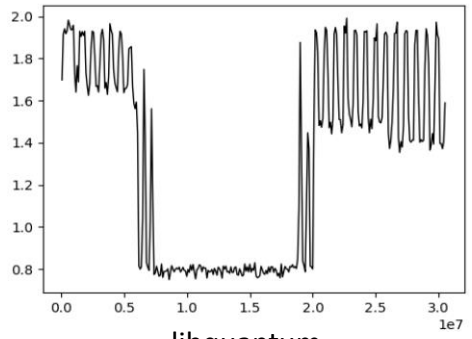




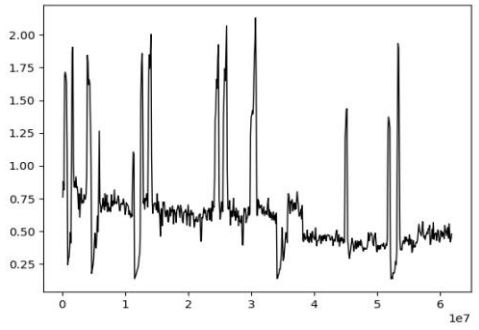
从**宏观**上发现香山**性能瓶颈**

XSPerfRolling 微观性能分析

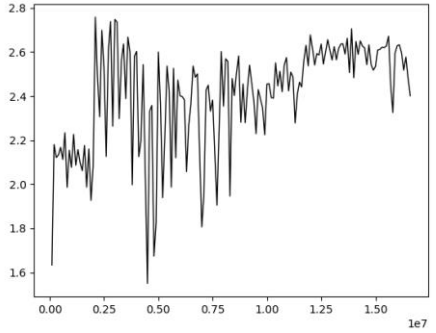
- **仿真运行各类 Workload 时，处理器性能表现往往不均一**
- 以若干 SPEC CPU 2006 Checkpoint 的 IPC rolling 曲线为例（40M cycles）：



libquantum

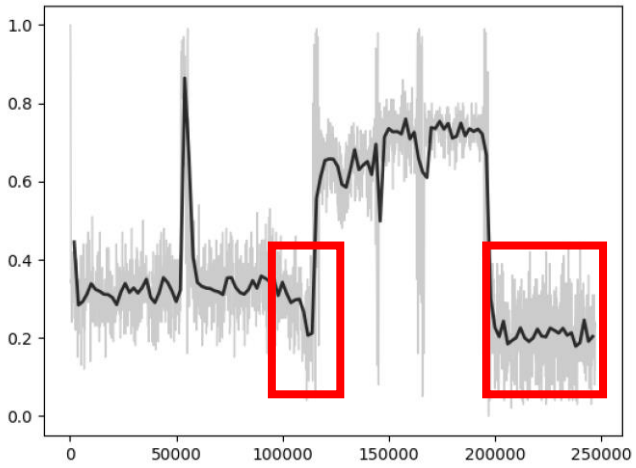


mcf

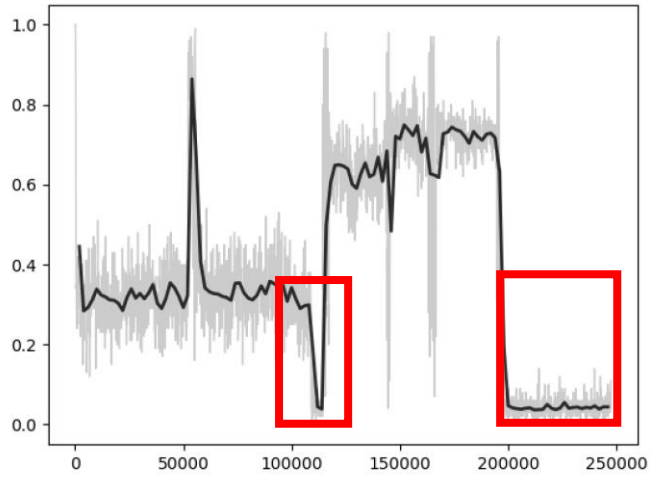


bwaves

- **一些架构/参数的改动只影响其中若干片断，性能计数器作为变化表征会被稀释**



Feature：预取器的训练时机改动

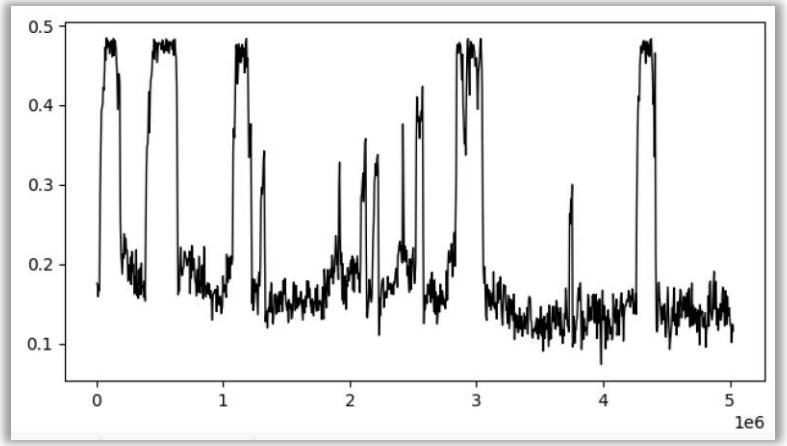


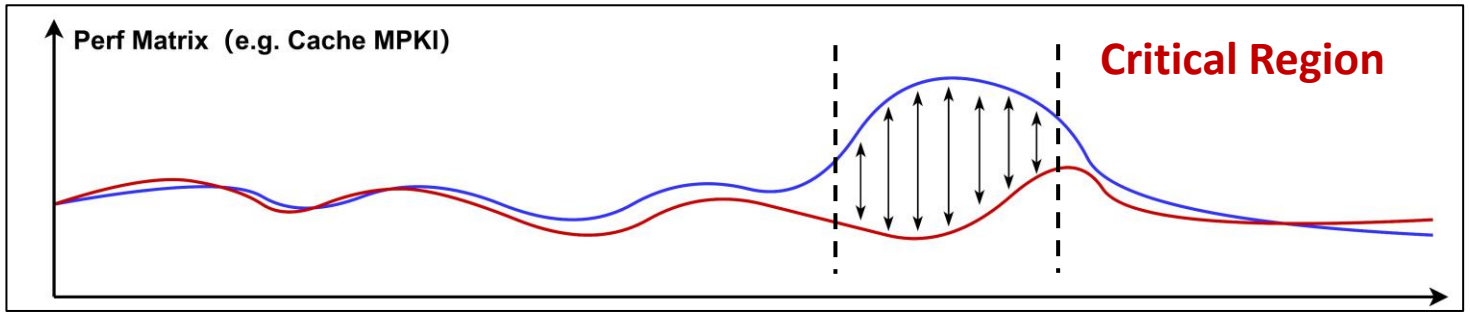
XSPerfRolling 微观性能分析

- **支持性能指标的 Rolling 曲线采集与可视化工具**
  - + 方便快捷的添加流程
  - + 可视化性能分析
  - + 精细分析片断间的性能差异
  - + 帮助锁定参数敏感的 Critical Region
  - - 存储开销更大 → **使用 ChiselDB 缓解存储问题**

```
XSPerfRolling(
  cacheParams, "L2PrefetchAccuracy",
  PopCount(l2prefetchUseful), PopCount(l2prefetchSent),
  1000, clock, reset
)
```

编译 + 仿真 + 脚本分析





如何更快地调整设计参数

方法1 → 参数 A 编译 → 仿真评估 → 参数 B 编译 → 仿真评估 → ……

问题①：编译与仿真串行化，参数调优效率低下

方法2 → 参数 A 编译 → 仿真评估

参数 B 编译 → 仿真评估

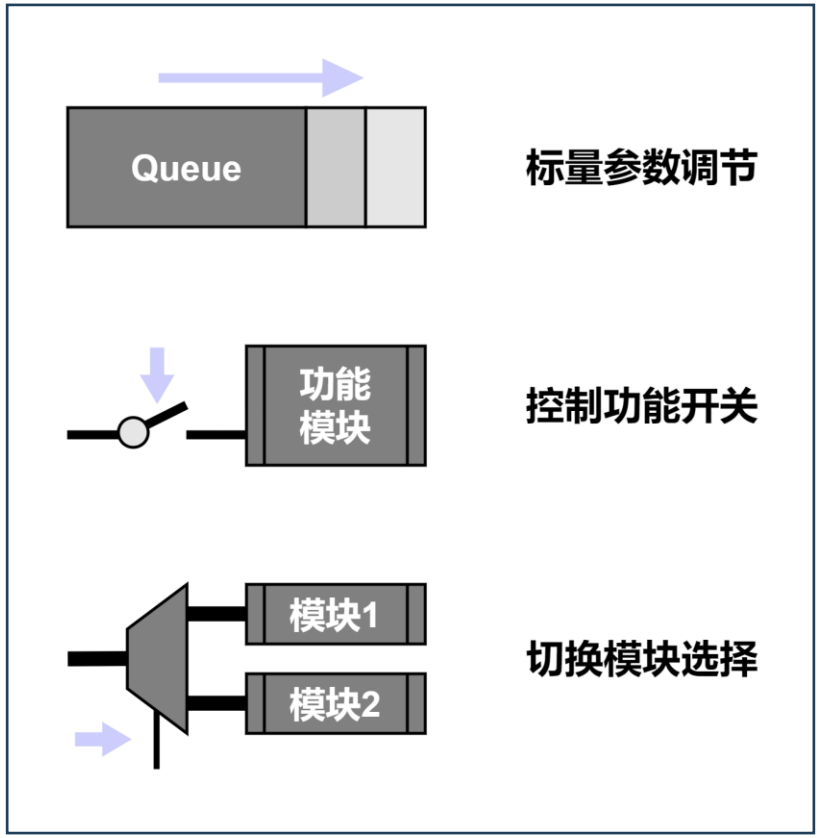
……

问题②：多次编译耗费大量服务器资源，扩展性不佳

新方法 → **使用 DPI 接口在运行时指定设计参数**

Constantin 运行时设计参数调控框架

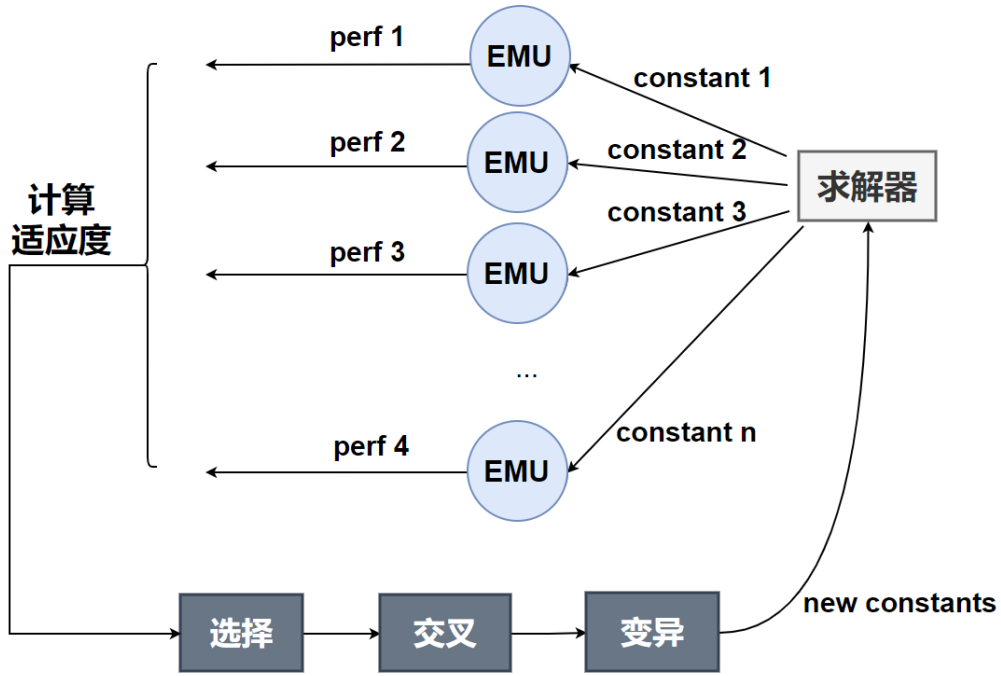
- **一种运行时的设计参数调控框架**
- **关键技术：用 DPI-C 接口指定参数**
  - **将确定常量取值的时刻从编译时推迟到运行时**
- **多种使用场景，敏捷多参数调控**
  - **控制标量参数**，如队列大小、请求延迟等
  - **控制功能开关**，如是否开启子分支预测器等
  - **切换模块选择**，控制Mux切换模块连线
- **优势**：**只需一次编译即可灵活调参，扩展性好**
  调参无需手动修改代码逻辑，解放人力



ConstantSolver 设计参数自动求解工具

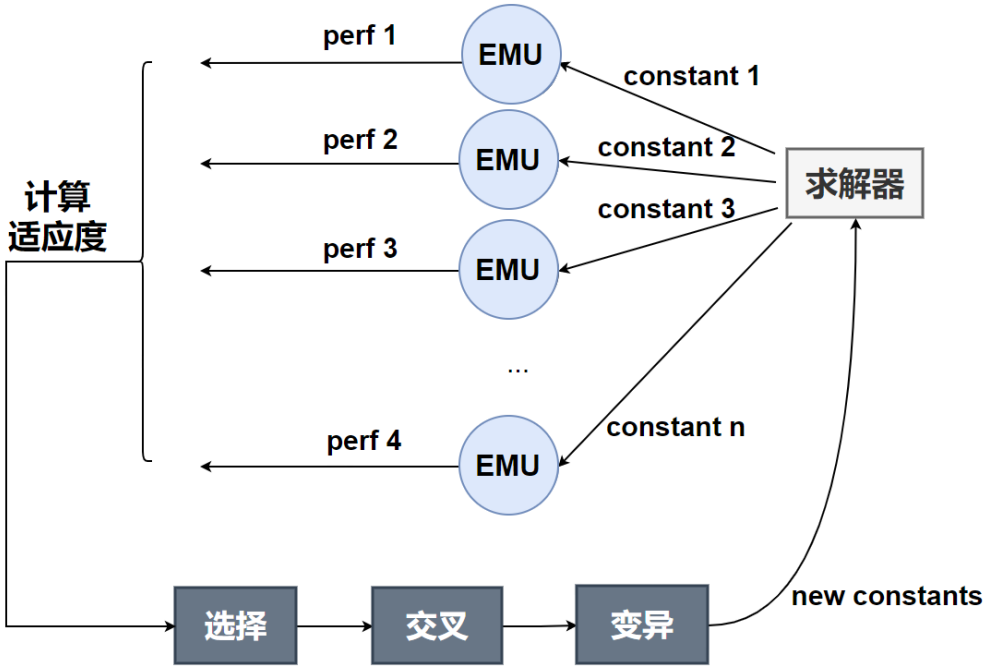
- **自动化迭代求解常量参数最优取值**
  - 基于 Constantin 运行时参数调控框架
  - 指定性能指标，仿真收集 RTL 性能计数值
  - 利用求解算法（如遗传算法）迭代常量参数值，并行迭代循环

优势：快速参数探索，解放调参所需人力



ConstantSolver 设计参数自动求解工具

- **设计参数求解真实案例**：求解 Load 指令 Cache miss 重发阻塞周期

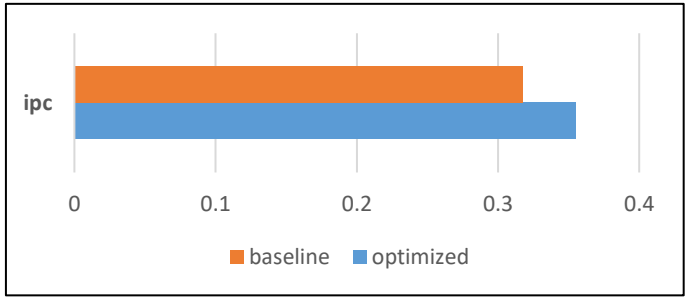


初始人工经验参数
（11，18，127，17）

求解目标：
最大化总线 forward 成功率
最小化缺失 penalty

自动求解参数结果
（7，0，126，95）

在访存敏感 Benchmark 上获得 11.6% 的性能提升



小结

- 香山高性能处理器开发进入深水区：**敏捷开发方法**助力快速迭代

- 弥补敏捷**功能验证**的短板：
  - 结构化信息辅助调试
  - 面向 RISC-V CPU 的覆盖率导向模糊测试

- 弥补敏捷**性能优化**的短板：
  - 从宏观到微观的性能分析流程
  - Constantin 运行时设计参数调控框架
  - ConstantSolver 设计参数自动求解工具

敏捷开发工具集在香山社区开源
https://github.com/OpenXiangShan