中国科学院软件研究所 Institute of Software Chinese Academy of Sciences
中国科学院计算技术研究所 INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES
北京开源芯片研究院 BEIJING INSTITUTE OF OPEN SOURCE CHIP

香山缓存系统的形式化验证

陈韬宇[1] 马锟[1] 陈熙[23] 王凯帆[23] 李勇坚[1] 蔡少伟[1]

[1]中国科学院软件研究所

[2]中国科学院计算技术研究所

[3]北京开源芯片研究院

2024年8月22日

报告内容

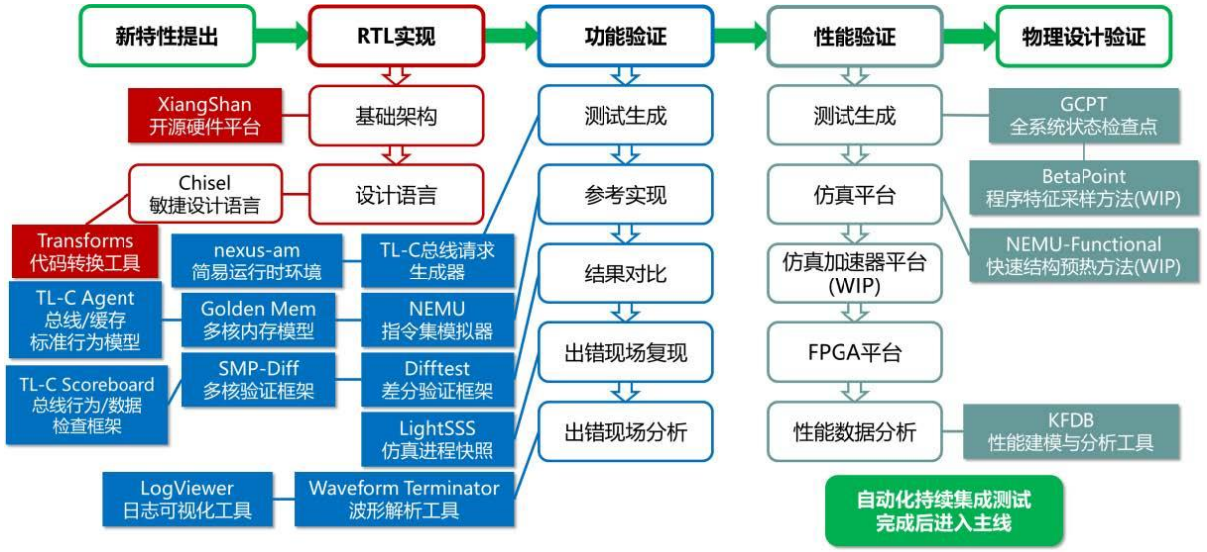
- 香山缓存与形式化验证
- 基于香山缓存的形式化验证方案
- 阶段性成果
- 未来研究方向

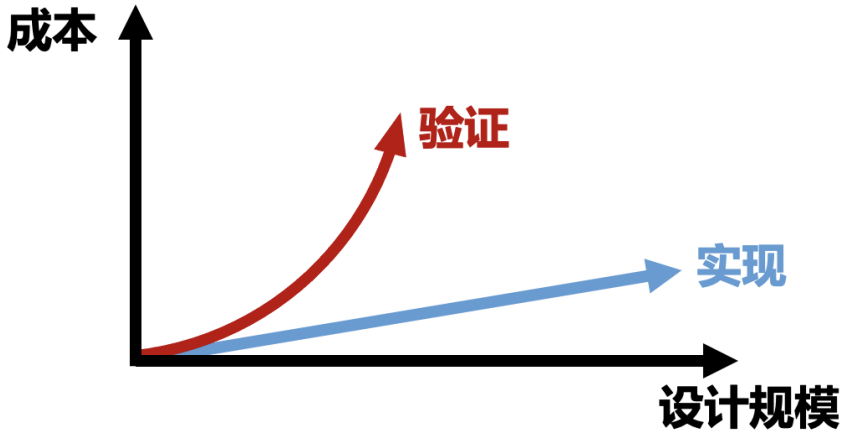
报告内容

- 香山缓存与形式化验证
  - 香山缓存系统介绍
  - 形式化验证：模型检测技术
- 基于香山缓存的形式化验证方案
- 阶段性成果
- 未来研究方向

香山缓存系统介绍

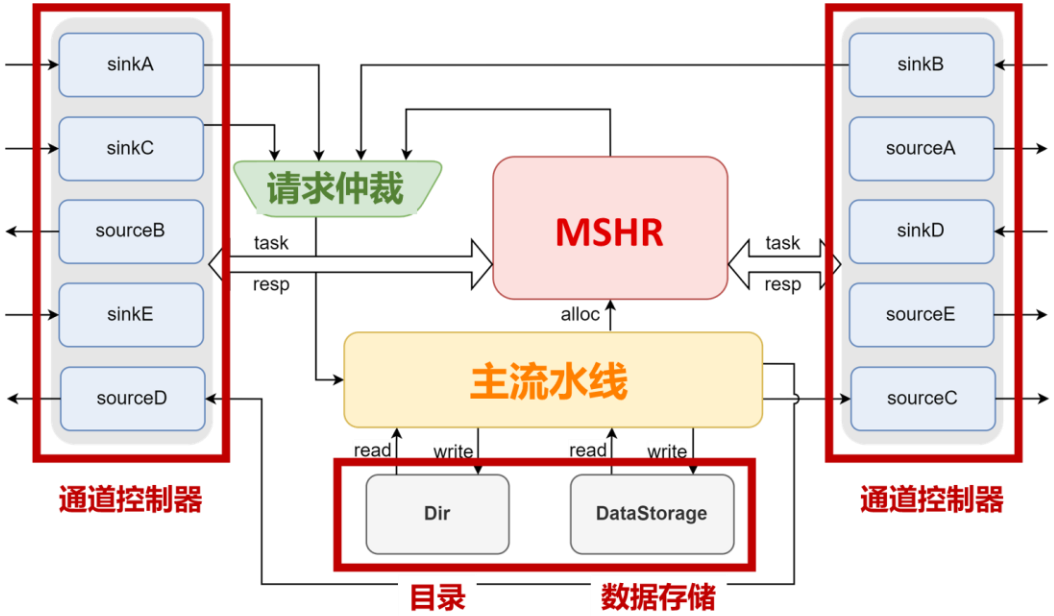
- 建立了包含设计、实现、验证在内全开源工具敏捷开发流程
- 采用 Rocket Chip 的 Diplomacy 框架实现组件互联
- 处理器设计与验证在敏捷度上的差距持续扩大，形成一堵验证墙



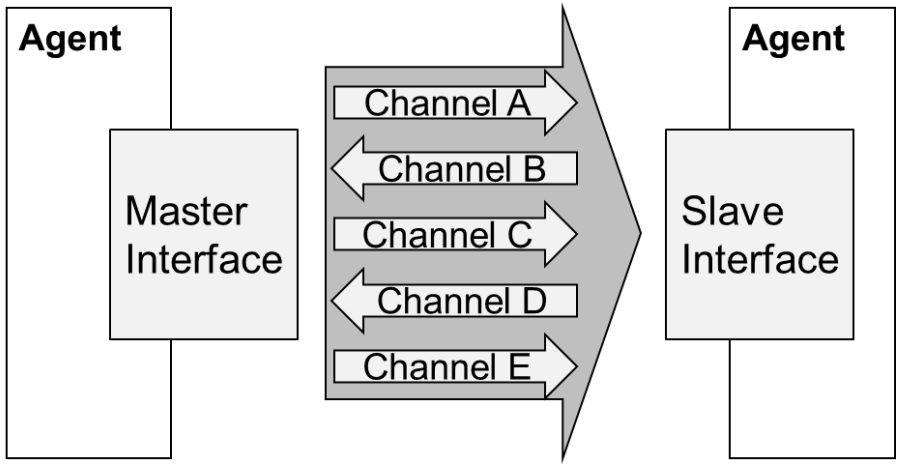


香山缓存系统介绍

- **HuanCun(南湖) → CoupledL2(昆明湖)**
  - 主流水线架构
  - Inclusive 策略
  - Directory – based
- **Diplomacy 框架**
  - 包括硬件实现和协商参数两部分
  - 编译期确定所有参数
  - 自动连接输入输出端口



CoupledL2 硬件结构



CoupledL2 实现的 TileLink 协议

香山缓存系统介绍

- **缓存系统所关注的一些性质**
  - 互斥性质：不同 cache 不能同时对同一地址的数据有写权限
  - 死锁性质：系统中的每一个请求都应当在有限时间内被响应

- **敏捷开发带来的验证困难**
  - 验证覆盖率提升困难：事务繁多复杂，难以找到极端的 Corner Case
  - 问题定位困难：大型项目结构繁杂，问题定位环节仍需要大量人力

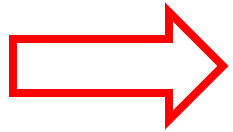
香山缓存系统与形式化验证

- **形式化验证**
  - 基于数学严格证明的自动化技术，可全面、精确地验证硬件电路设计满足预定规范
  - **完备性、可重复性、高效**
- **缓存系统所关注的一些性质**
  - 互斥性质：各缓存的状态需满足特定条件
  - 死锁性质：各请求应在有限时间内被响应

香山缓存系统与形式化验证

- **形式化验证**
  - 基于数学严格证明的自动化技术，可全面、精确地验证硬件电路设计满足预定规范
  - **完备性、可重复性、高效**

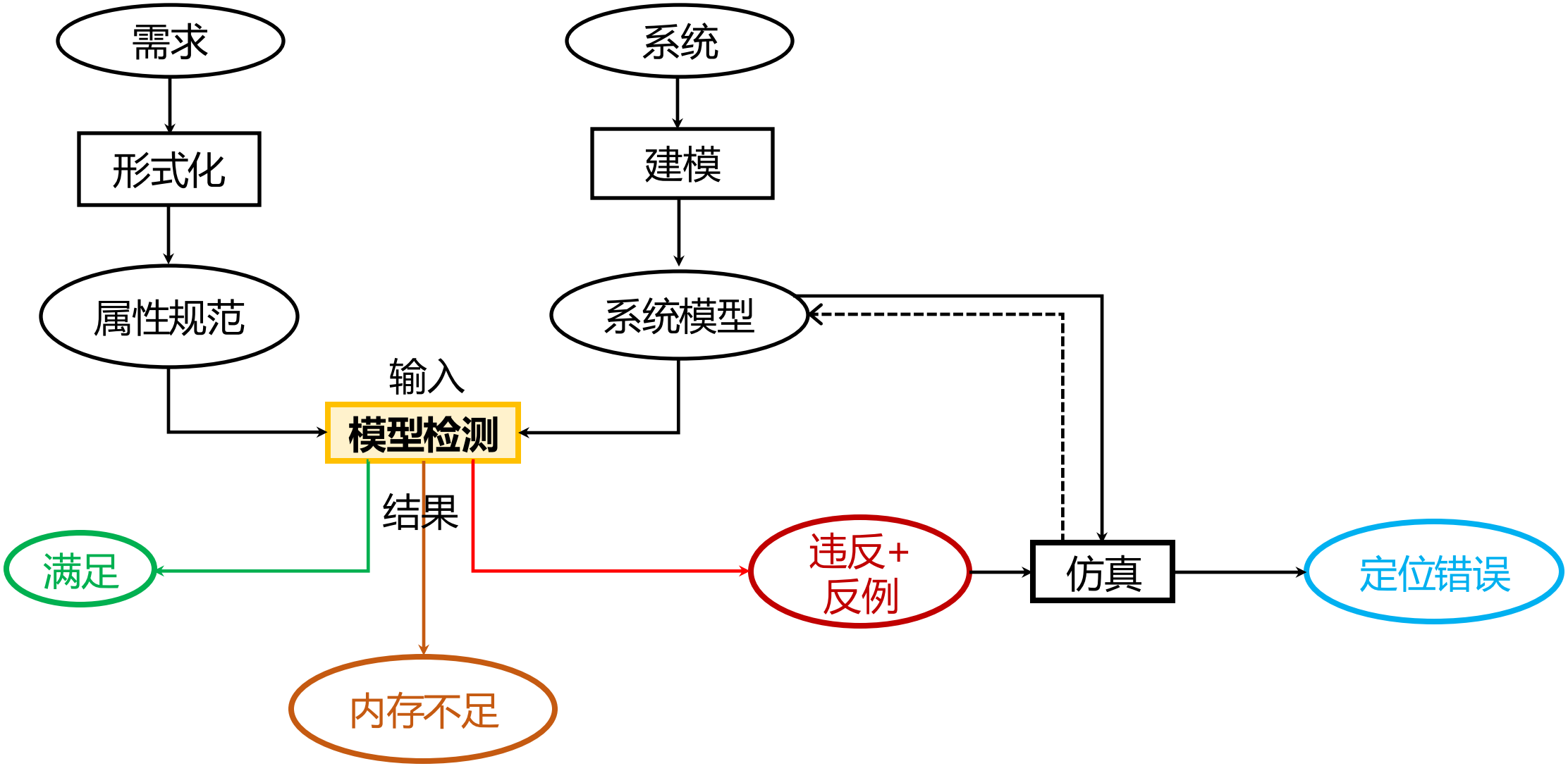
- **缓存系统所关注的一些性质**
  - 互斥性质：各缓存的状态需满足特定条件
  - 死锁性质：各请求应在有限时间内被响应



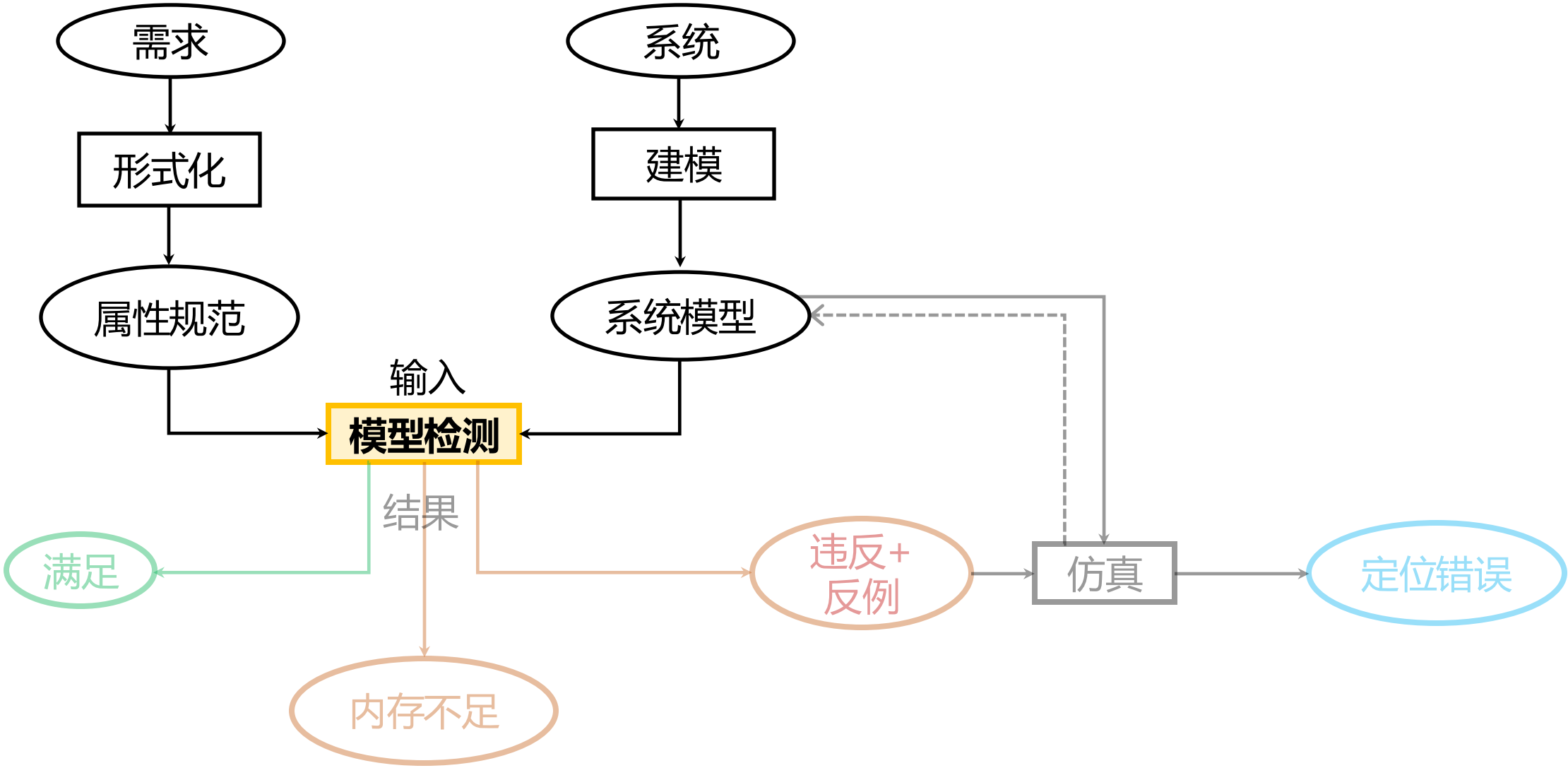
典型安全性质（Safety）

典型活性性质（Liveness）

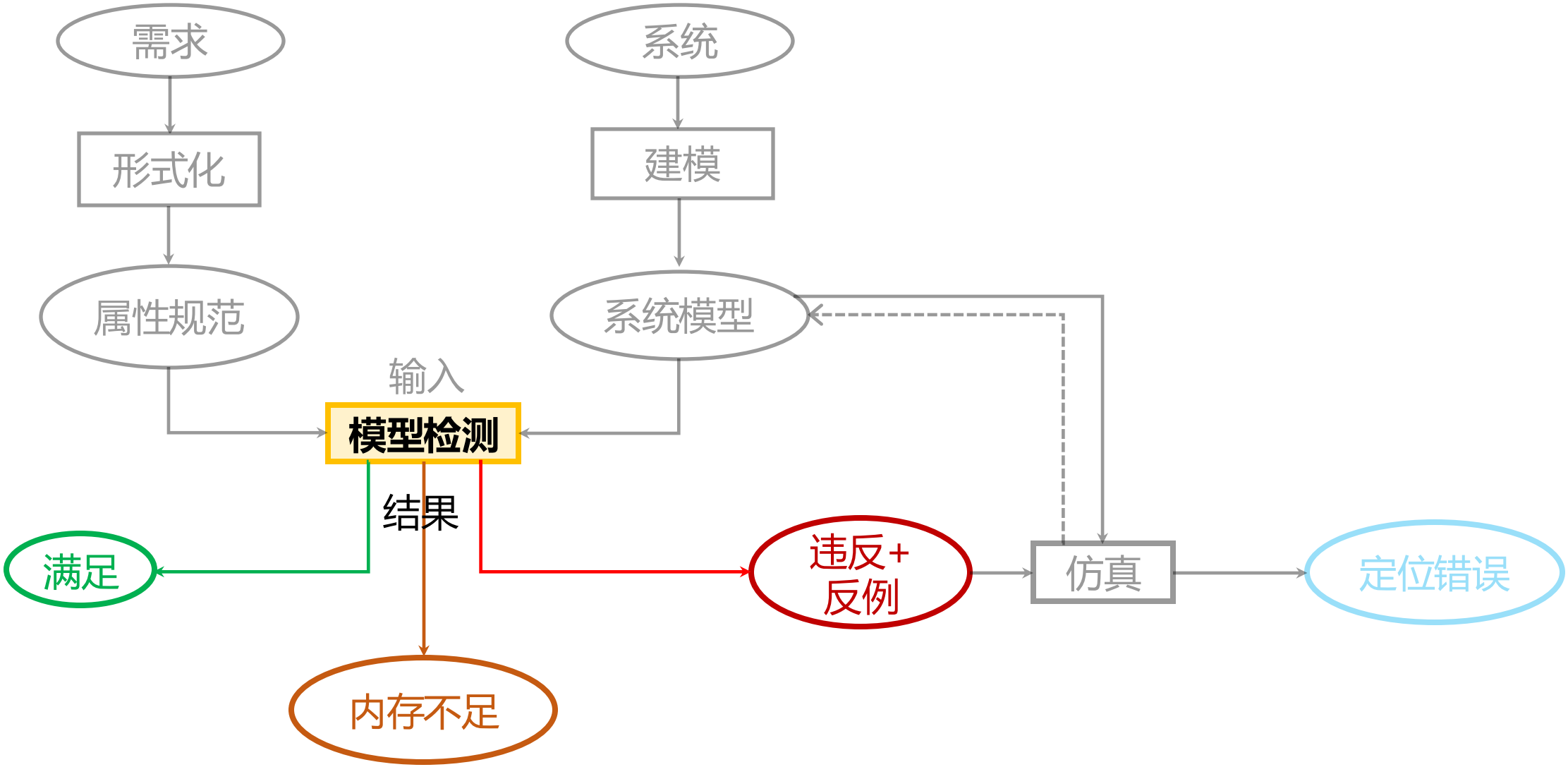
形式化验证技术之一：模型检测



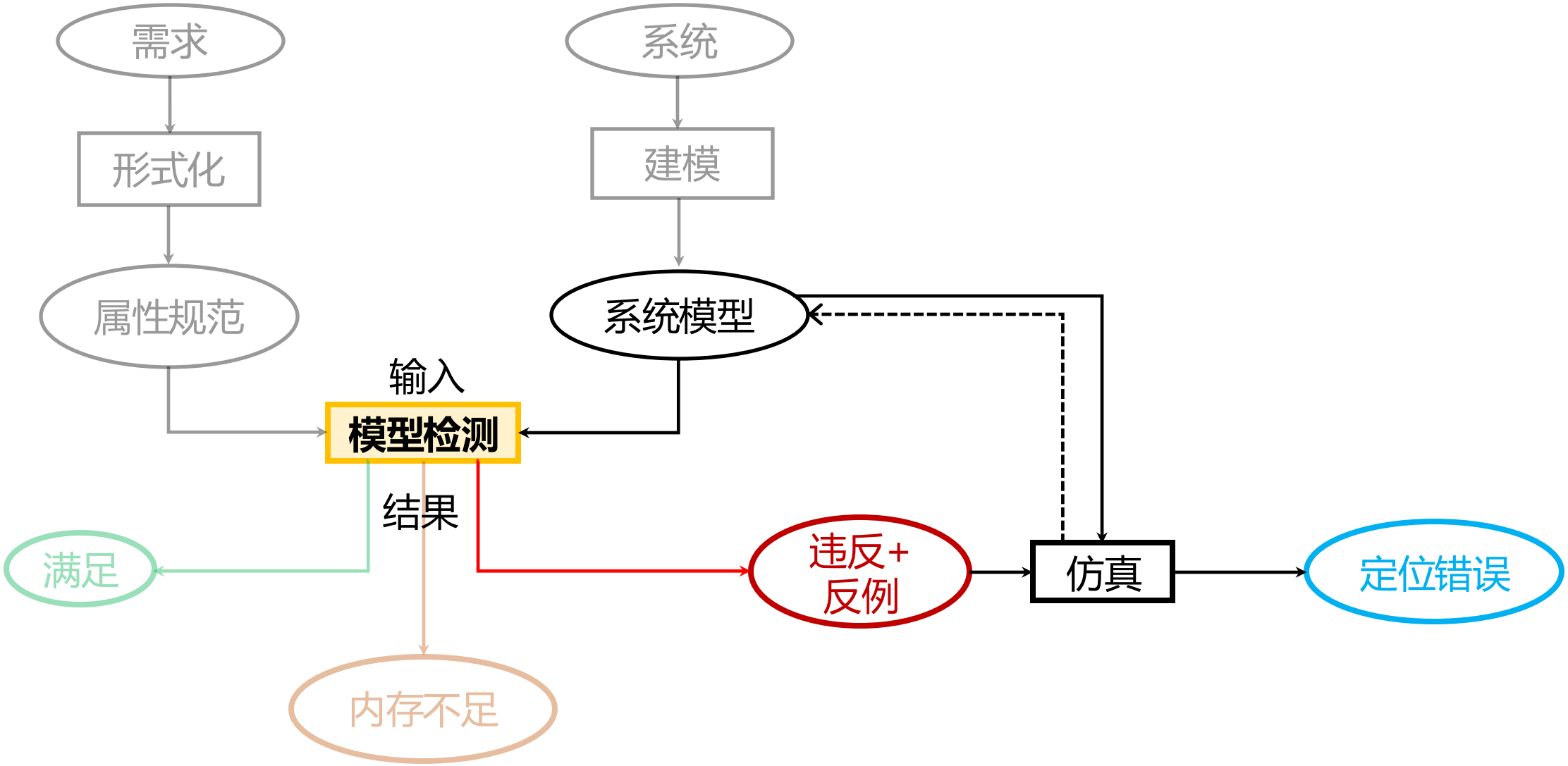
形式化验证技术之一：模型检测



形式化验证技术之一：模型检测



形式化验证技术之一：模型检测

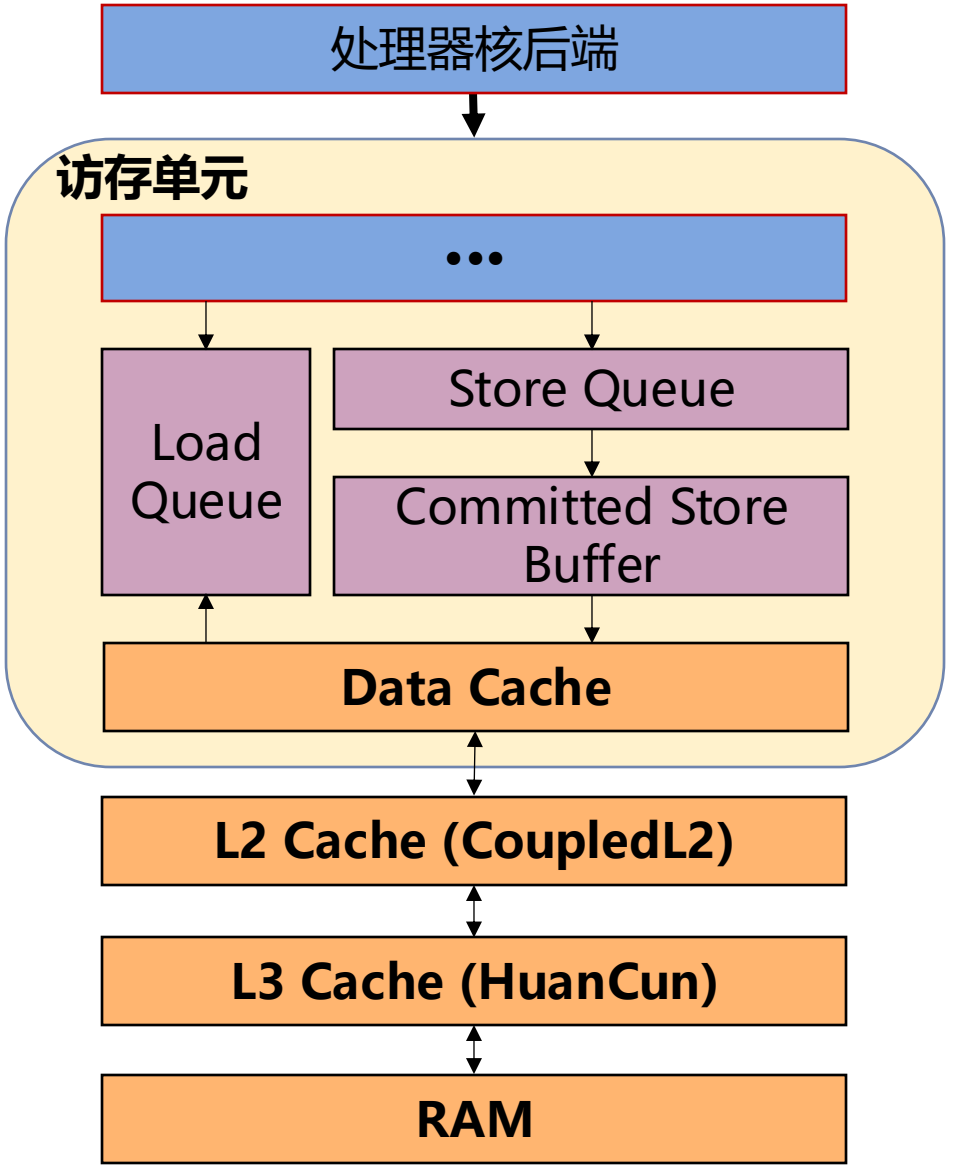


报告内容

- 香山缓存与形式化验证
- 基于香山缓存的形式化验证方案
  - 香山形式化验证的难点
  - 系统建模与性质定义
  - 验证加速方案
- 阶段性成果
- 未来研究方向

香山形式化验证的难点①

- **缓存系统联系紧密**
  - 完整的缓存系统涵盖三级缓存和访存流水线，难以将L2、L3缓存隔离验证
  - 缓存接口无法对接形式化验证接口



香山缓存系统示意图

香山形式化验证的难点②

- **并发事务复杂**

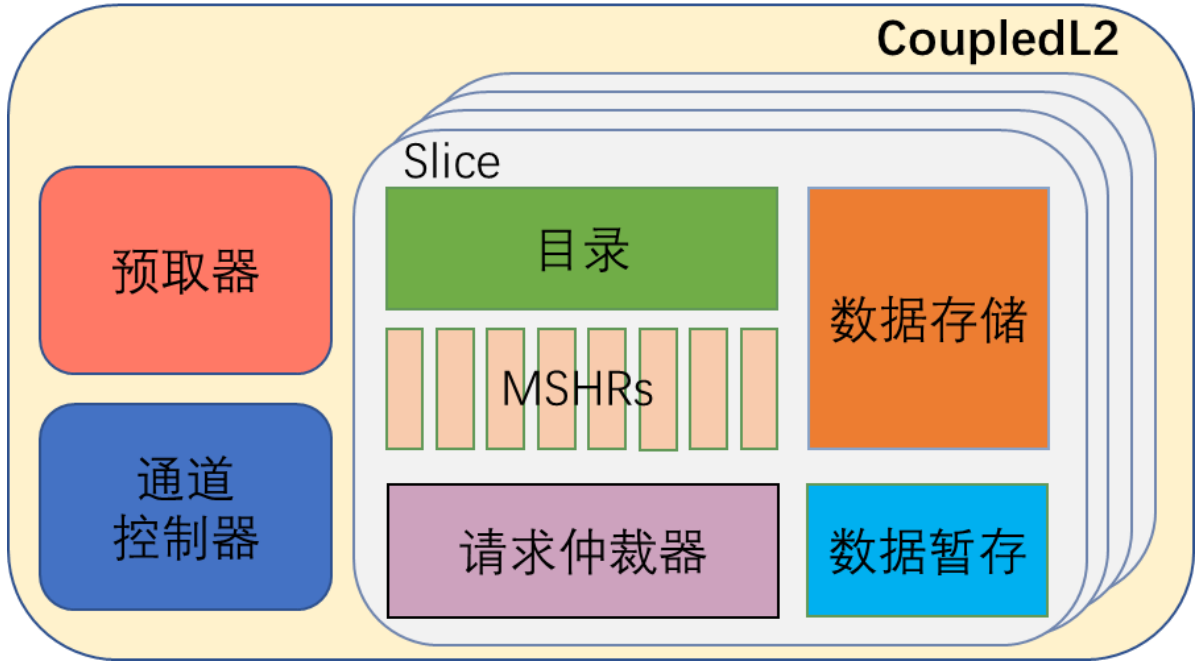
多处理器核并行

缓存内多Slice并行

Slice内多MSHR并行

} 状态空间指数级增加

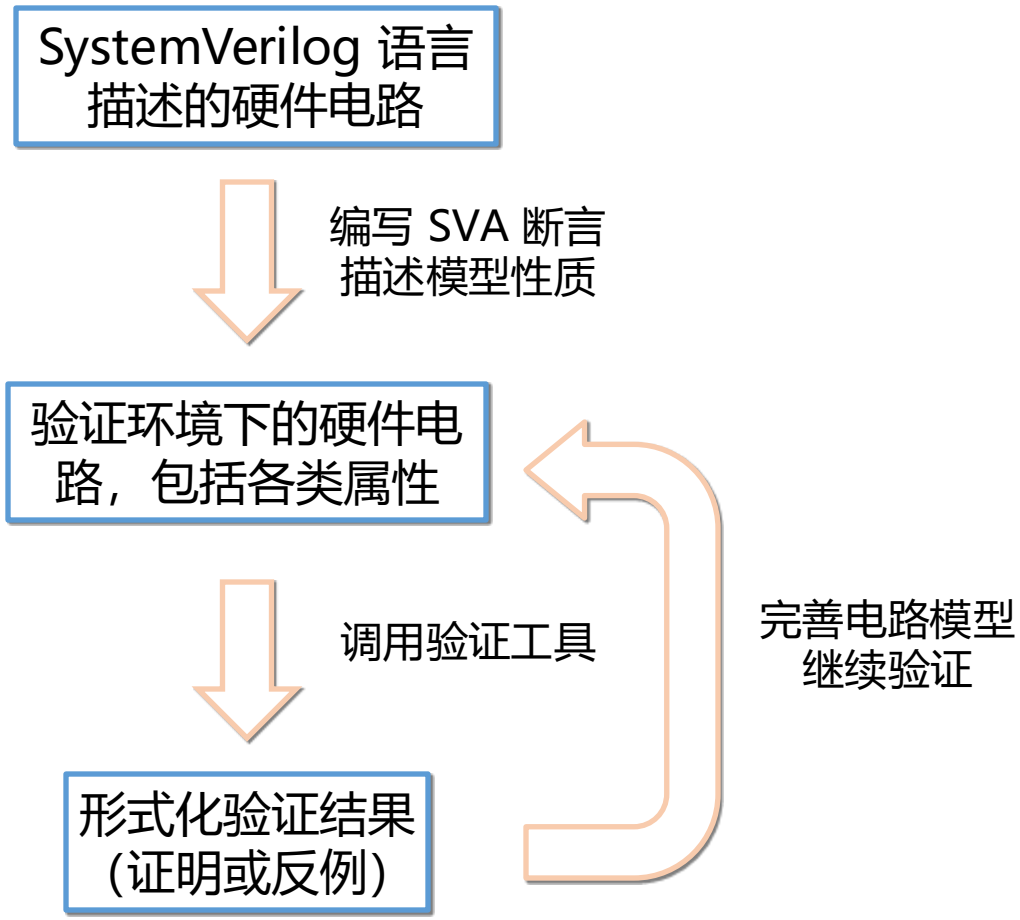
- 复杂错误极难找到，且不易复现



CoupledL2 缓存架构示意图

香山形式化验证的难点③

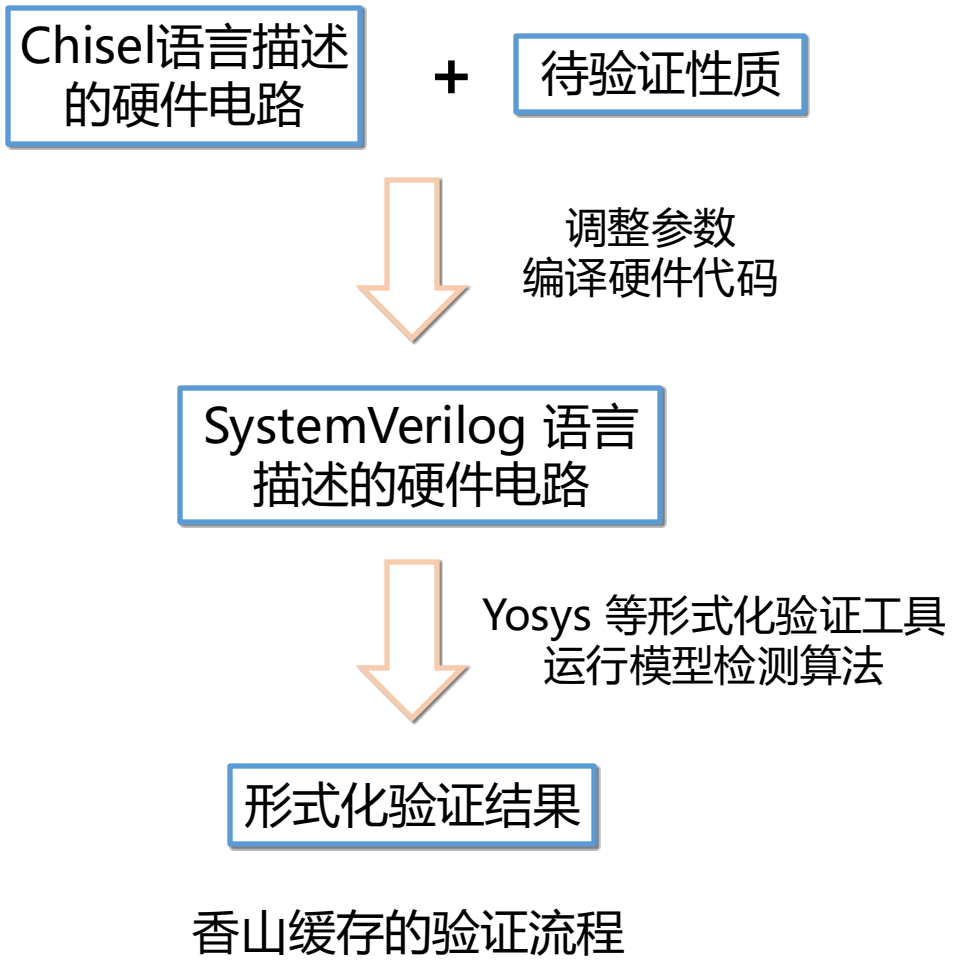
- **形式化验证工具链不成熟**
  - 敏捷开发环境下对 Chisel 语言设计的形式化支持不够
  - 硬件领域的大多数形式化验证生态建立在Verilog、SystemVerilog等语言上



SystemVerilog项目的典型验证流程

香山缓存的形式化验证方案

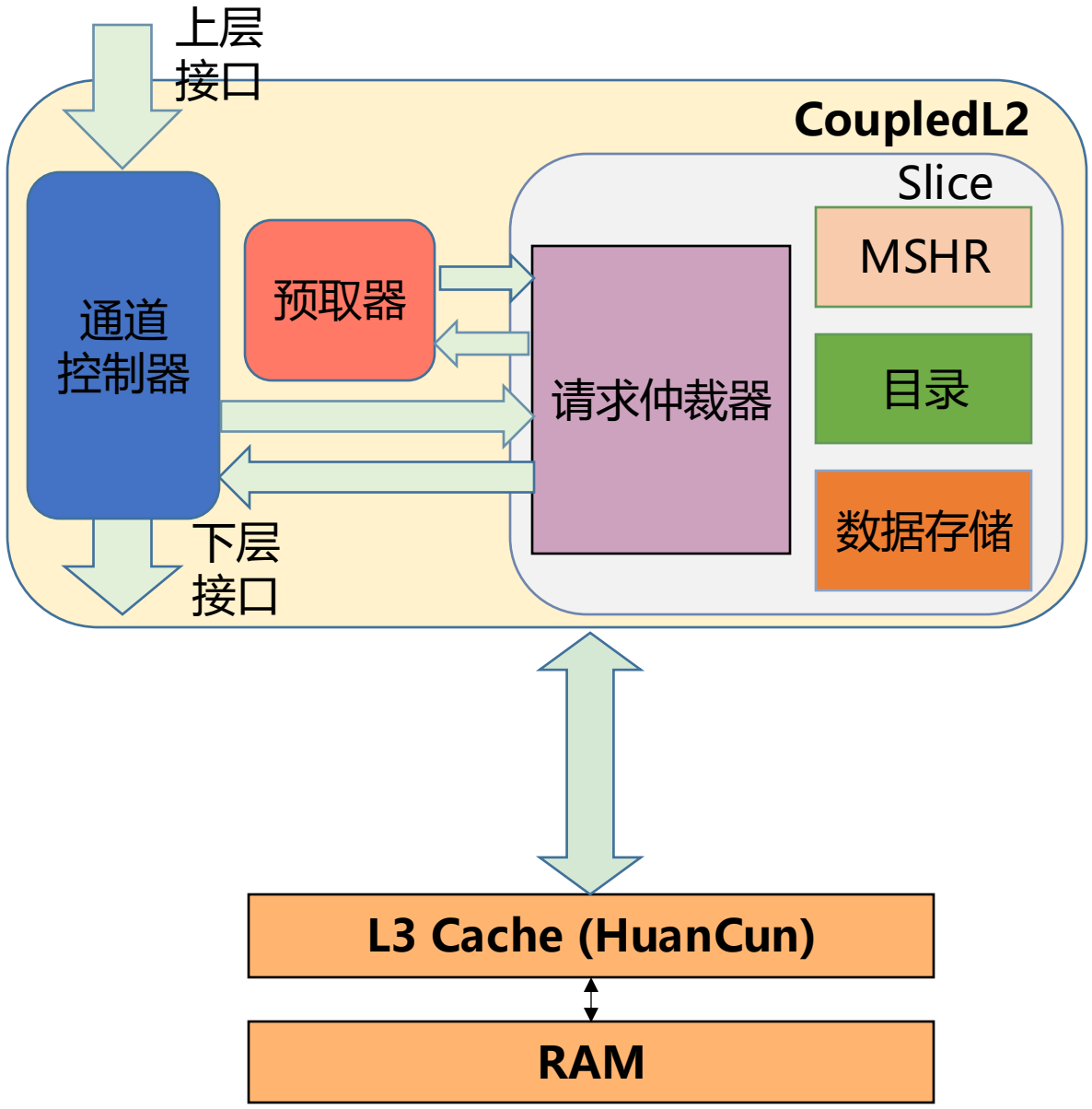
- **验证流程**
  - 设计辅助模块，建模待验证的性质
  - 调整验证参数，使电路规模适合验证环境
  - 导出 Verilog 代码，运行验证算法
- **结果反馈**
  - 判定给定的性质是否被满足
  - 提供反例波形图，记录触发性质的场景



香山缓存的验证流程

香山缓存的形式化验证方案

- **L1 缓存及接口设计**
  - 验证环境下需要 L1、L2、L3 和 RAM 独立运行，由外部输入控制
  - 以 CoupledL2 本身为基础改造出验证环境的 L1 缓存 （**CoupledL2AsL1**）



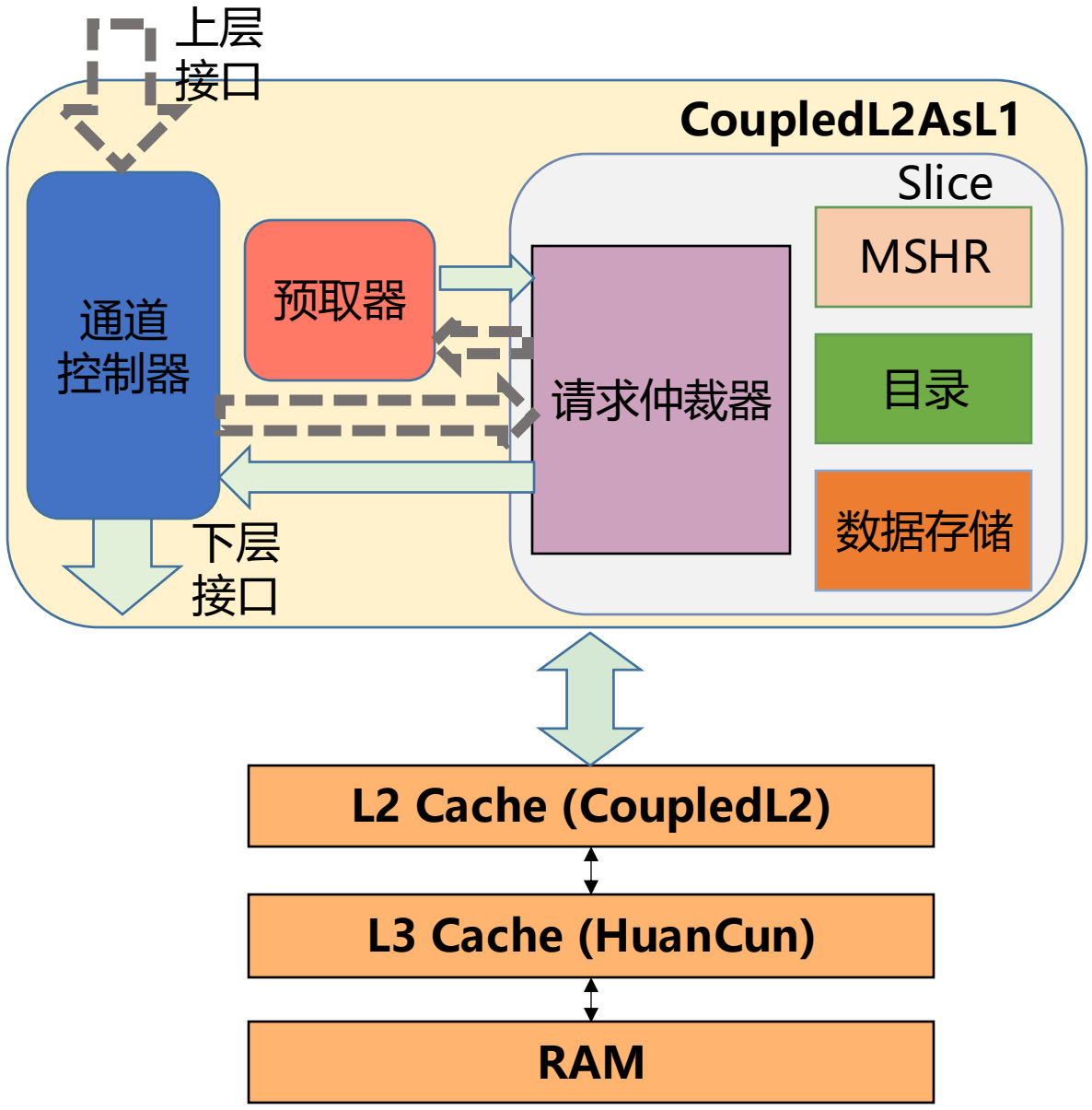
CoupledL2 事务示意图

香山缓存的形式化验证方案

- **L1 缓存及接口设计**
  - 以 CoupledL2 本身为基础改造出验证环境的 L1 缓存 （**CoupledL2AsL1**）

1. 通过**预取器**而非上层接口控制访存

2. 添加形式化**验证接口**



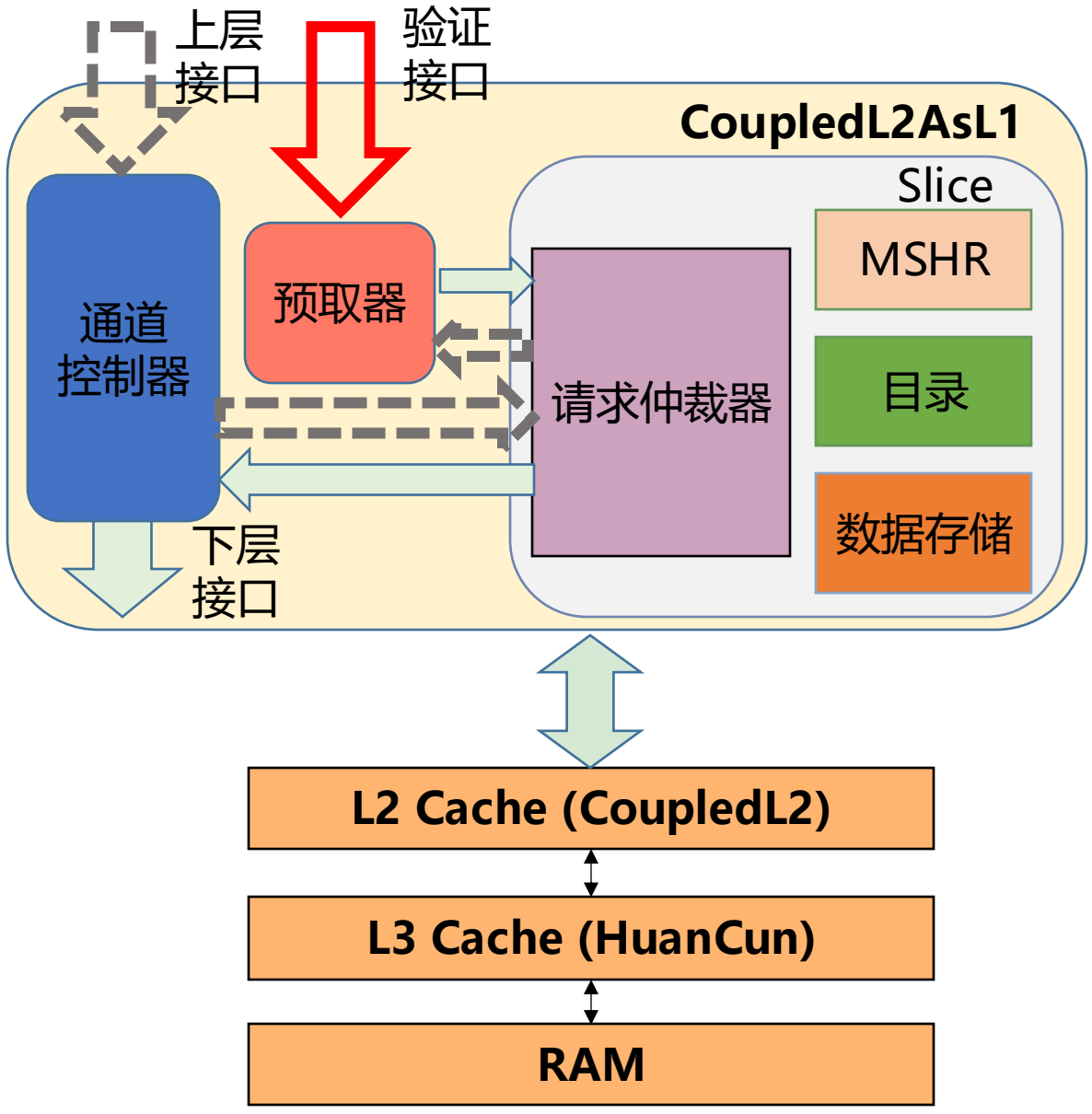
CoupledL2AsL1 事务示意图

香山缓存的形式化验证方案

- **L1 缓存及接口设计**
  - 以 CoupledL2 本身为基础改造出验证环境的 L1 缓存 （**CoupledL2AsL1**）

1. 通过**预取器**而非上层接口控制访存

2. 添加形式化**验证接口**



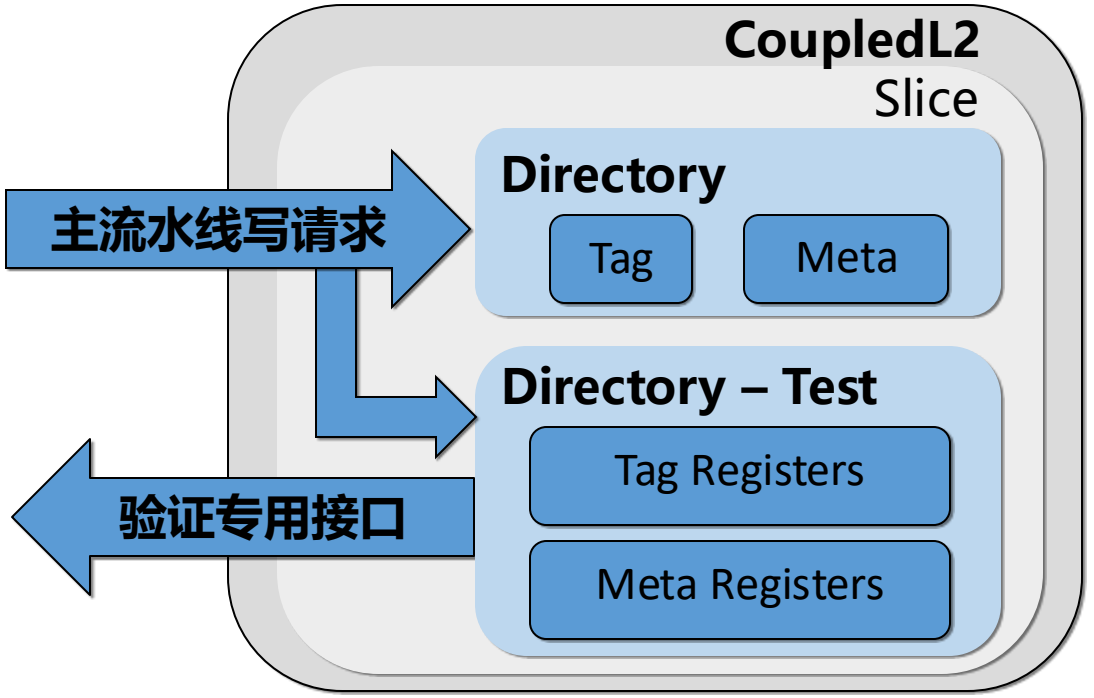
CoupledL2AsL1 事务示意图

香山缓存系统建模

- **待验证系统**
  - 香山 L1(**CoupledL2AsL1**) + L2(**CoupledL2**) + L3(**HuanCun**) + RAM
  - 约 2 万行 Chisel 代码 → 18 万行 Verilog 代码 → 极大规模电路
- **所关注性质**
  - 互斥性质：各 L2 缓存内对应同一地址的缓存行状态须符合 TileLink 协议
  - 死锁性质：L1 缓存发送的每一个请求都应当在有限时间内收到对应的响应

形式化验证：互斥性质定义

- **背景**：香山缓存的缓存行状态位于目录模块，使用 SyncReadMem 存储
- **难点**：
  - 目录模块无法随时读取多个缓存行
  - 验证逻辑会影响事务正常运行
- **方案**：
  - 编写寄存器阵列模拟目录模块，与原模块接入相同的写请求
  - 读接口专用于验证环境，可随时读取任意地址缓存行



目录辅助模块示意图

形式化验证：死锁性质定义

```
property check_deadlock;
  @ (posedge clock) disable iff(~reset)
    (nodeOut_a_ready ##[1:$]
    (nodeOut_d_valid &
     nodeOut_d_bits_source == nodeOut_a_bits_source));
endproperty
assert property(check_deadlock);
```

```
always @(posedge clock) begin
  if (resetCounter_notChaos & ~reset)
  begin
    assert(deadlockTimer < 64'h2710);
  end
end
```

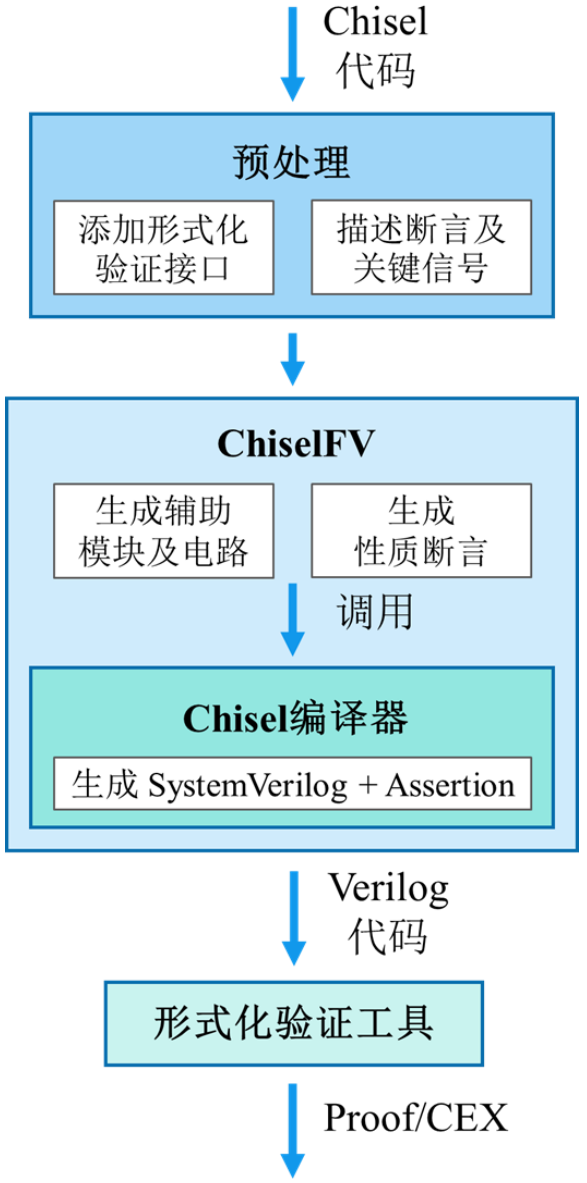
L1 发出请求后无穷时刻都不被响应，则视为发生死锁

L1 请求 10000 拍后仍不被响应，则 L2 可能出现了死锁

	Liveness Property	基于 Timer 的 Safety Property
验证流程复杂度	较高	较低
后端算法的验证效率	较低	较高
反例的可阅读性	较低	较高

形式化验证：打通 Chisel – Verilog 工具链

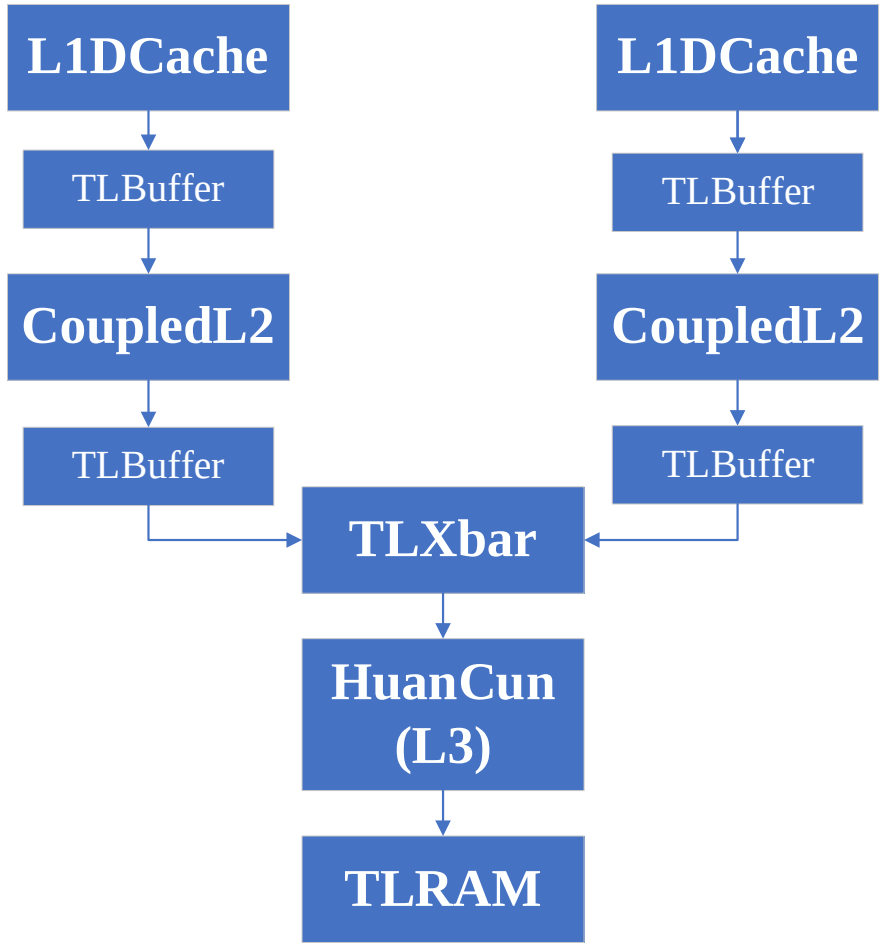
- **背景：**暂无形式化验证工具支持 Chisel 语言
- **难点：**
  - Chisel 导出的 Verilog/SystemVerilog 代码可读性极低
  - Chisel 语言暂无成熟的形式化验证插件（基于Chisel v3.6.0）
- **方案：ChiselFV 工具**
  - 在 Chisel 层面设计验证接口，由工具生成辅助电路
  - ChiselFV 同时生成形式化验证工具的定制脚本，实现自动化
- **成果：**"ChiselFV: A Formal Verification Framework for Chisel." *DATE 2023*



ChiselFV 工具链示意图

加速香山缓存验证：建立精简模型

- **简化系统结构**
  - 保证功能完备的同时组件最少
  - 构造单 L3(HuanCun) + 双 L2(CoupledL2) + 双 L1(CoupledL2AsL1) 的**层次化**香山缓存系统



验证环境系统结构示意图

加速香山缓存验证：建立精简模型

- **削减参数规模**
  - 保证性质等价，保留一定并行度和有效结构的同时参数最小
  - 验证环境使用能独立运行、组相联、多MSHR并行的**最小三级缓存单元**

参数名称	缓存组件	削减前	削减后
ways	L2	8	2
	L3	16	
sets	L2	512	4
	L3	4096	
banks	L2/L3	4	1
blockBytes	L2/L3	512	2
busWidth	L2/L3	256	1
mshrs	L2	16	4
	L3		6
address	RAM	36位	5位

验证环境参数表

加速香山缓存验证：添加 assume 语句

- 限制搜索空间，尽可能精细地引导工具寻找反例路径
- **重要代码片段：**

```
for(i <- 0 until nrL2) {
  assume(io.topInputValids(i) ||
    (io.topInputRandomAddrs(i) === 0.U &&
     io.topInputNeedT(i) === false.B))
}
```

- 输入的激励无效(!io.topInputValids)时，保持其他验证变量不变
- 从输入端最大程度地削减搜索树

报告内容

- 香山缓存与形式化验证
- 基于香山缓存的形式化验证方案
- 阶段性成果
  - 香山缓存错误场景：互斥性质
  - 香山缓存错误场景：死锁性质
- 未来研究方向

阶段性成果

- 主代码仓库：

  OpenXiangShan/CoupledL2 (github.com)

- 验证版本：
  - master 分支 commit 514c1ad – *2023.11.01*
  - master 分支 commit c974407 – *2024.04.02*
  - master 分支 commit 9e841f3 – *2024.06.19*

- **互斥性质：** 发现香山某一版本设计疏漏，事务逻辑不符合 TileLink 协议要求
- **死锁性质：** 发现多个重要死锁场景并修复
- 均已得到香山开发团队确认

断言编号	状态组合 (根据 TileLink 协议)	是否触发	期望结果
1 - 4	Tip - Tip	NO	NO
5 - 8	Tip - Trunk	NO	NO
9 - 12	Tip - Invalid	NO	NO
13 - 16	Branch - Branch	YES	YES
17 - 20	**Tip - Branch**	**YES**	**NO**
21 - 24	Trunk - Invalid	YES	YES

互斥性质验证结果

性质描述	断言编号	验证结果	波形长度	验证时间
互斥性质	VerifyTop.assert_v0	**proven**	infinite	**1842.482 s**
	VerifyTop.assert_v1	**cex**	487	**80.328 s**
	VerifyTop.assert_v2	**undetermined**	84 -	**252135.649 s**
死锁性质	mshrCtl.assert_v0	**cex**	1099	**21.516 s**
	mshrCtl.assert_v1	**cex**	8050	**152.253 s**
	mshrCtl.assert_v2	**cex**	5156	**1623.286 s**
	mshrCtl.assert_v3	**cex**	14994	**20325.492 s**
	mshrCtl.assert_v4	**undetermined**	1061 -	**143020.065 s**

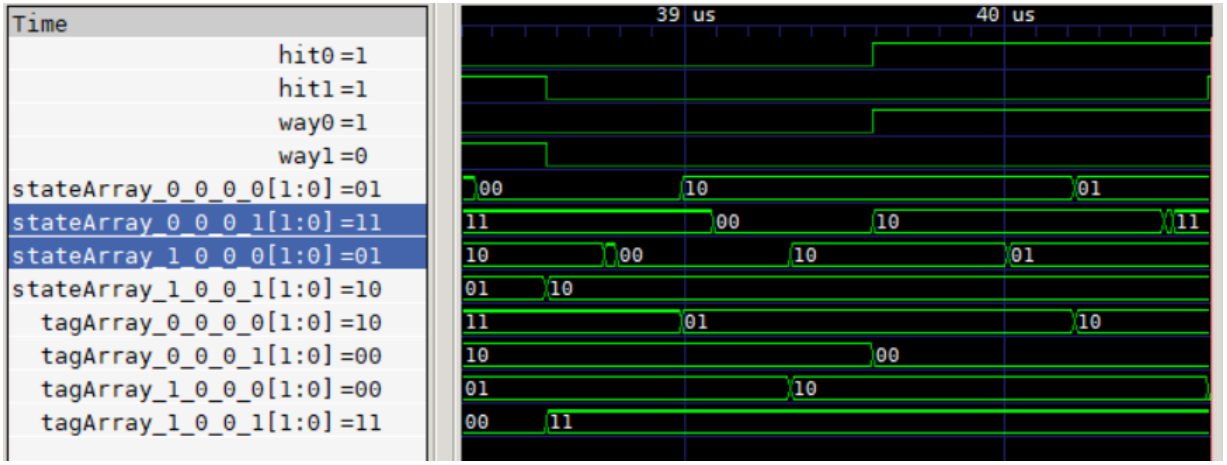
各版本详细验证结果

香山缓存错误场景分析：互斥性质

触发性质

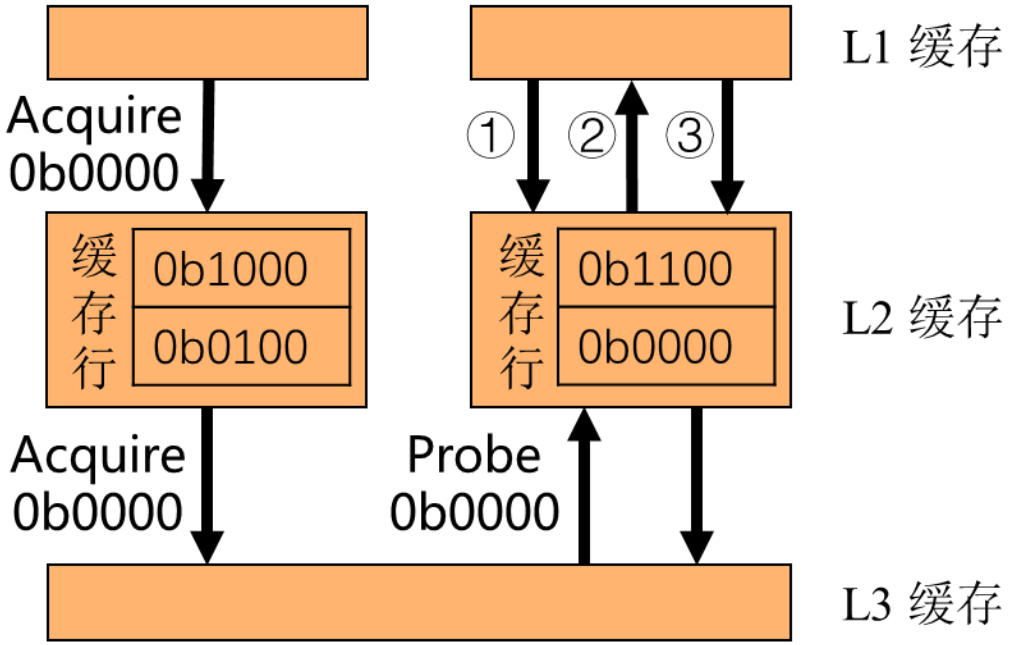
两个 L2 缓存中同一地址对应的缓存行状态出现不符合 TileLink 协议的组合： Tip – Branch

对应波形图



触发原因分析

①: Release TtoN
②: Probe TtoB
③: ProbeAck NtoN

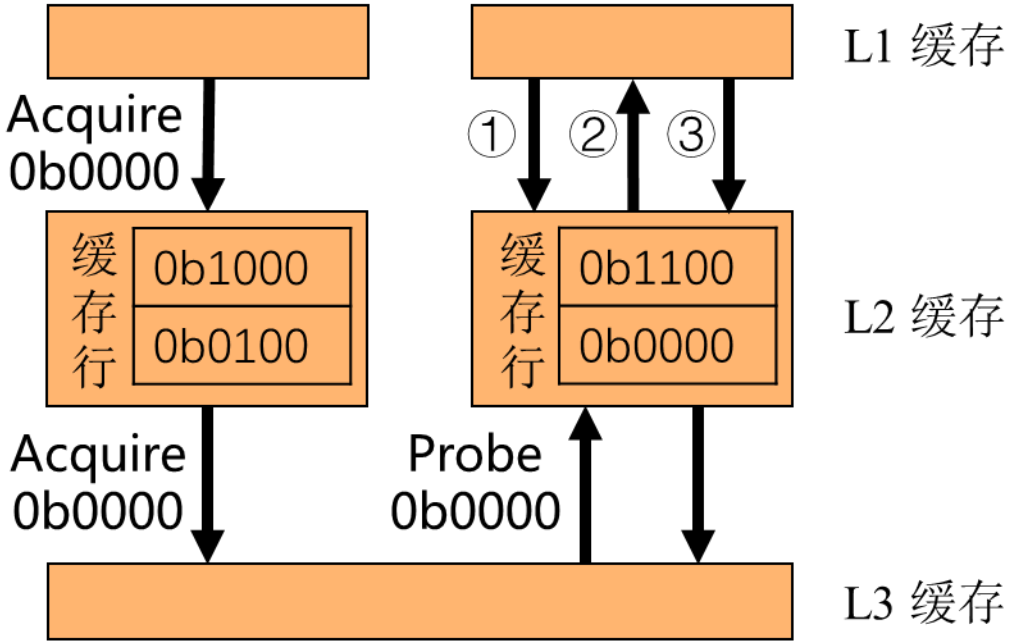


根据 TileLink 协议，L1 需要在收到 ReleaseAck 之后才能发送 ProbeAck，而香山缓存未实现这一细节

香山缓存错误场景分析：互斥性质

修复方法

①: Release TtoN
②: Probe TtoB
③: ProbeAck NtoN



代码提交

Commit

SinkB: fix bug for not accept Probe when same-addr waiting ReleaseAck...

…, split the replaceConflict conditions (#208)

Co-authored-by: cai luoshan <cailuoshan@node005.boscluster.com>

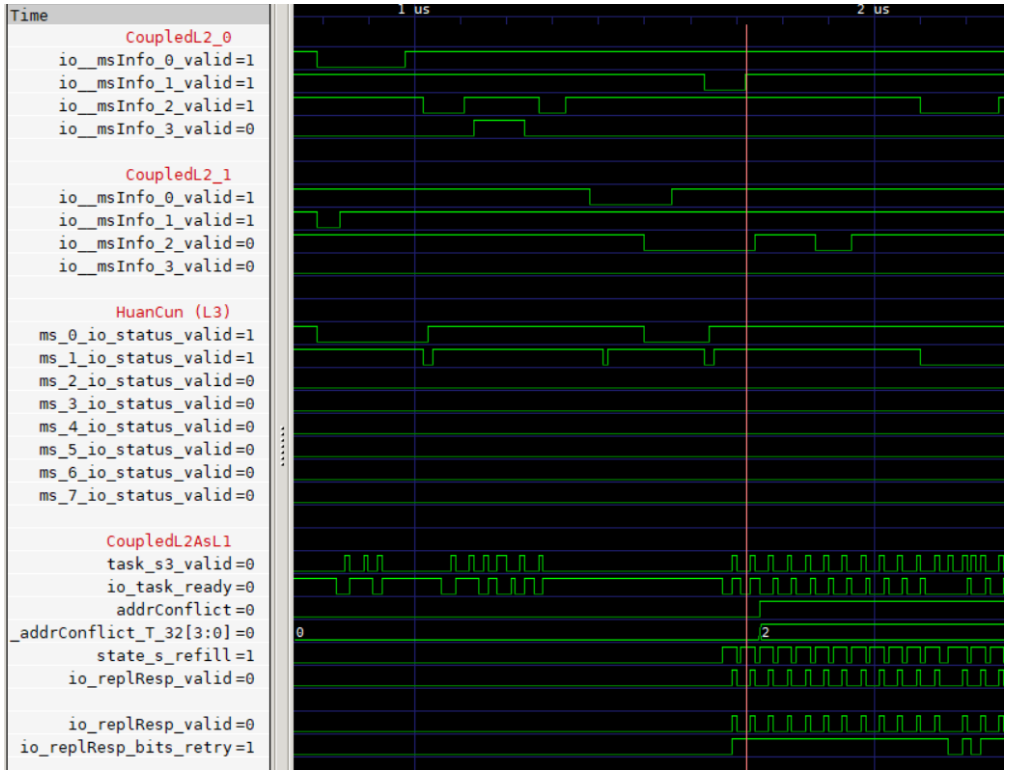
master (#208)

cailuoshan and cai luoshan committed 2 weeks ago Verified

- 修改发送 ProbeAck 的条件，保证其在收到 ReleaseAck 之后才发送 ProbeAck

香山缓存错误场景分析：死锁性质

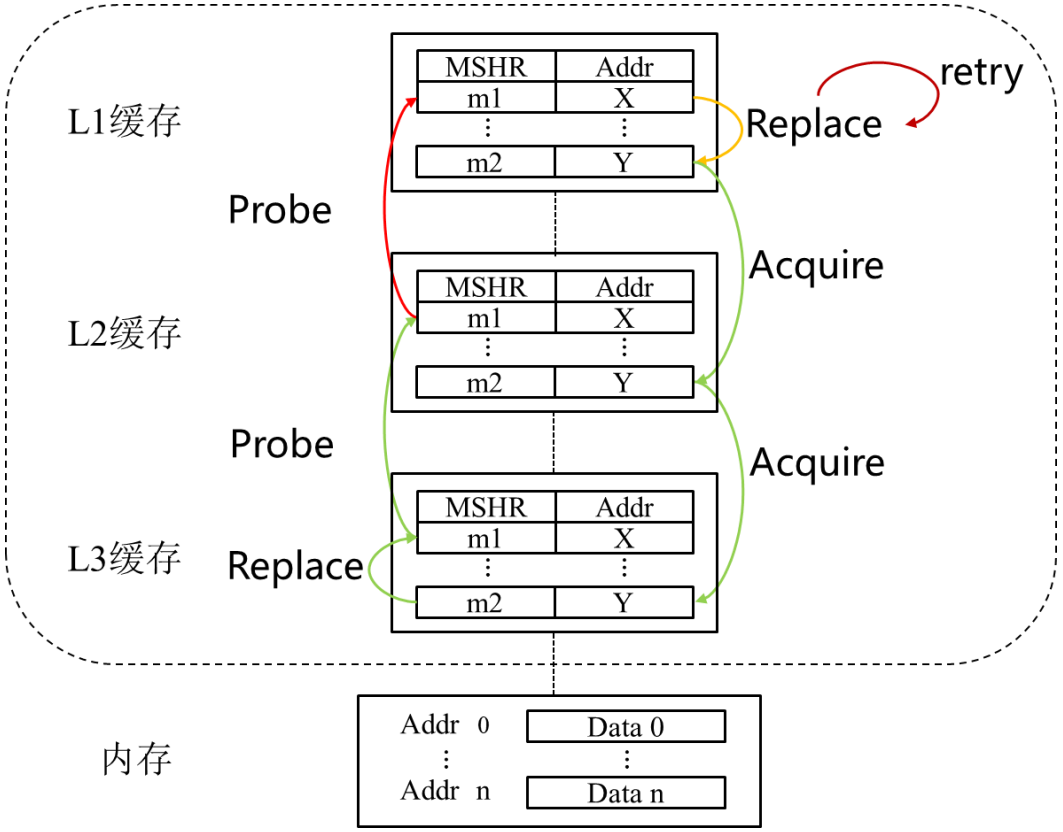
对应波形图



L1缓存的替换被阻塞，不断重试

- 这一死锁验证方法已得到计算所、开芯院认可

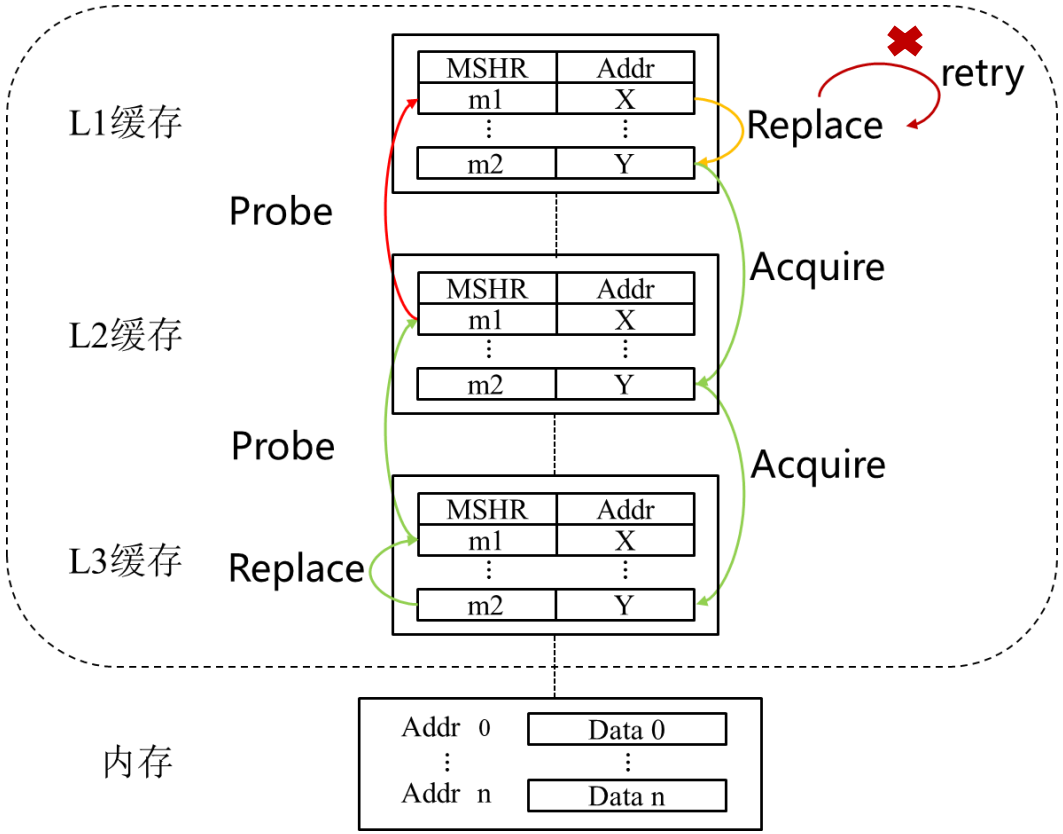
死锁原因分析



CoupledL2 不恰当的替换策略导致了死锁

香山缓存错误场景分析：死锁性质

修复方法



代码提交

Commit

✓ Retry wayMask: fix dead lock bug when mshr_refill retry (#83)

```
* Retry wayMask: fix dead lock bug when mshr_refill retry, choose another way

* Retry wayMask: fix bug for way update

---------

Co-authored-by: Cai Luoshan <cailuoshan18@mails.ucas.ac.cn>
```

master (#83)
v0.9

cailuoshan and Cai Luoshan committed on Dec 11, 2023 Verified

- 修改替换策略，多次失败时尝试替换其他路

报告内容

- 香山缓存与形式化验证
- 基于香山缓存的形式化验证方案
- 阶段性成果
- 未来研究方向

未来研究方向

- 关注更多的重要缓存性质，更全面地验证缓存系统
  - inclusive 性质、数据一致性性质
- 敏捷地生成 Formal Testbench 以及相应的断言
  - 提高验证敏捷度
- 将工具抽象封装成可复用的接口，方便泛化和移植

团队介绍：软件所蔡少伟研究团队

研究方向

- 约束求解
  - 逻辑约束求解：SAT，SMT，MaxSAT
  - 运筹优化：混合整数线性规划， 非线性规划
- EDA形式化验证
  - 等价性验证
  - 模型检查
  - Chisel验证平台
  - 缓存协议验证

代表性成果

- 多次获得SAT比赛金牌，SMT比赛金牌，MaxSAT比赛冠军
- 是近两年来Z3求解器的主要代码贡献者，研发了Z3++
- 刷新了MIPLIB测试集中多个 open实例的求解记录
- 在CAV，FM，DAC，ICCAD，SAT，CP， AAAI ，Artificial Intelligence，ACM Trans. on Computational Logic，IEEE Trans. on Computers等顶级会议和期刊发表约百篇论文
- 获得CAV，SAT，CP等领域顶级会议的最佳/杰出论文奖
- 成果应用于华为公司的OS内核和EDA多个场景，包括进入华为鸿蒙创新大赛决赛的优秀项目（2024进行中）
- 求解器和EDA技术应用于多家EDA企业，以及iEDA开源平台
- 运筹优化求解器应用于阿里妈妈的合约广告，微软公司的云计算预配置，以及各种信息通讯，能源调度等场景

……

谢谢！
敬请批评指教！