



# 面向服务器的香山处理器 开源多核解决方案

丁昊楠<sup>1</sup> 张林隽<sup>2</sup> 陈熙<sup>1</sup> 蔡洛姍<sup>1</sup> 王凯帆<sup>1</sup> 袁宇翀<sup>1</sup>

朱昱<sup>2</sup> 马久跃<sup>2</sup> 张睿思<sup>1</sup> 马月骁<sup>1</sup> 郑楚育<sup>2</sup>

<sup>1</sup>中国科学院计算技术研究所 <sup>2</sup>北京开源芯片研究院

2024年8月22日



# 目录

- 背景
- 开源生态
  - 验证方法
  - 硬件 IP
- 解决方案
  - 过去 (TileLink)
  - 未来 (AMBA CHI)
- 总结



# 目录

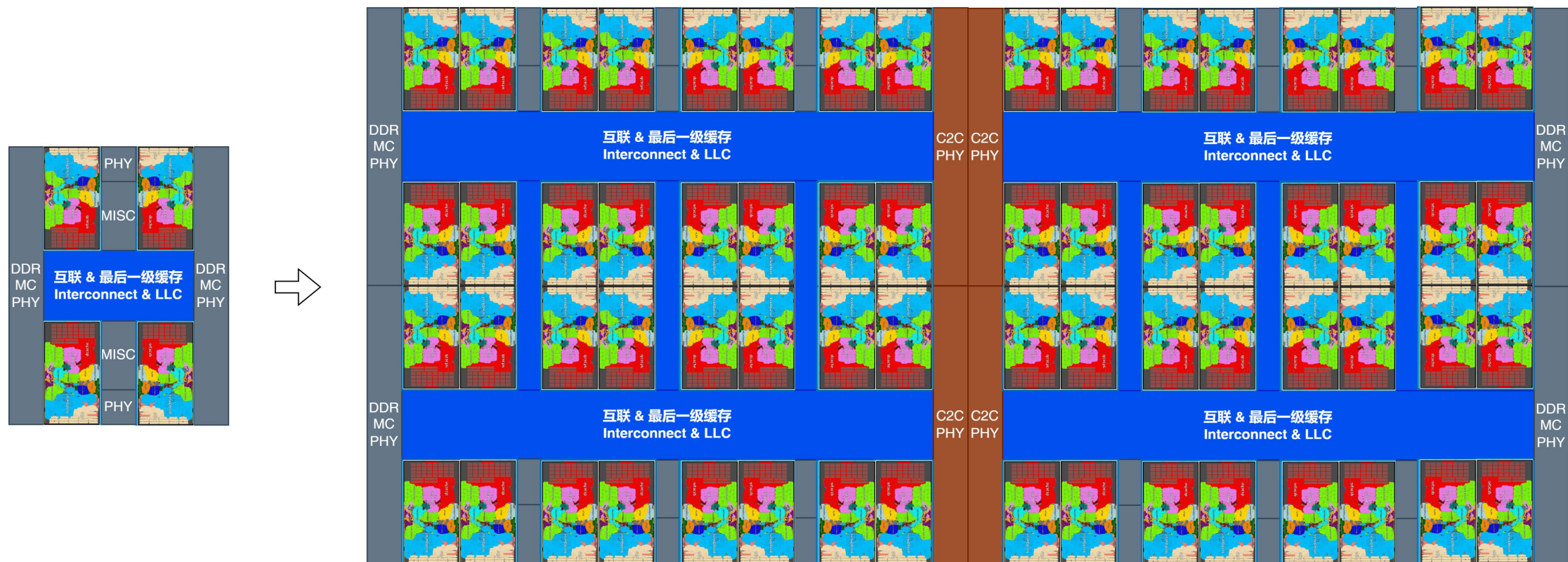
- 背景
- 开源生态
  - 验证方法
  - 硬件 IP
- 解决方案
  - 过去 (TileLink)
  - 未来 (AMBA CHI)
- 总结

# 背景

## → 服务器平台的处理器需要更多核心

- **并行计算需求增加**
  - 多任务处理
  - 高并发需求
- **虚拟化和云计算**
  - 计算资源分配
  - 弹性规模扩展
- **数据密集型应用兴起**
  - 大数据分析
  - 机器学习和人工智能
- **多线程编程的普及**
  - 单核性能进步放缓
  - 开发者的转变
- **能源效率与热管理**
  - 功耗控制
  - 相对较低的频率
- **技术进步**
  - 制造工艺的提升
  - 总线架构优化

## → 扩展更多核心的关键在于互联总线





# TileLink vs AMBA CHI

| 特性                                 | TileLink    | AMBA CHI |
|------------------------------------|-------------|----------|
| 一致性状态                              | 类 MESI      | MOESI    |
| 流控                                 | Ready-valid | Credit   |
| 拓扑感知                               | 无           | 有        |
| QoS                                | 无           | 有        |
| 内存序支持                              | 未定义         | 有(支持多种)  |
| 电源、复位、时钟管理                         | 无           | 有        |
| Cache-to-cache<br>Direct Transfer  | 未定义         | 有        |
| Cache-to-memory<br>Direct Transfer | 未定义         | 有        |
| CMO                                | 无           | 有        |
| Exclusive                          | 无           | 有        |
| Atomic                             | 有           | 有        |

支持复杂总线拓扑 (如 Mesh)  
的必要标准化定义

支持复杂总线性能优化  
的必要附加功能定义

支持一致性操作性能优化  
的必要附加功能定义

## 小结 ①

**TileLink 总线不能满足服务器平台的产品化需求**



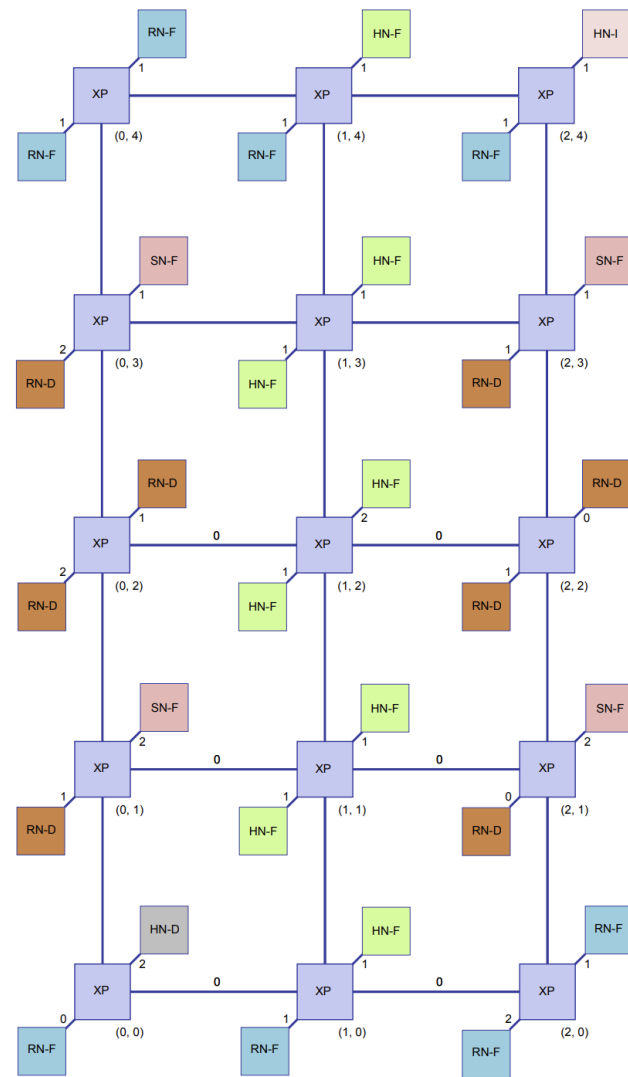
# 目录

- 背景
- **开源生态**
  - 开发工具
  - 硬件 IP
- 解决方案
  - 过去 (TileLink)
  - 未来 (AMBA CHI)
- 总结



# 🌟 开源互联总线生态——工具

- 开发 = 设计 + 测试
  - 计算平台复杂度不断提高
    - 多核、多级缓存、异构
    - 高性能计算终将拥抱 NoC
  - 缓存系统的复杂度不断攀升
    - 高并发度的硬件实现
    - 拥有最多 Corner Case 的部件之一
- 基础设施工具直接决定开发效率
- 缓存一致性验证的重要性和复杂度不断提高



ARM CMN600 3x5 Mesh NoC 示例

# 🌟 开源互联总线生态——工具

| 需求      | TileLink         | AMBA CHI (过去) |
|---------|------------------|---------------|
| 协议层抽象   | 有 (TL-Test)      | 无             |
| 事务级抽象   | 有 (TL-Test)      | 无             |
| 一致性检查   | 有 (TL-Test)      | 无             |
| 测试环境构造  | 有 (TL-Test)      | 无             |
| 可约束随机测试 | 有 (TL-Test)      | 无             |
| 快速测例构造  | 有 (TL-Test)      | 无             |
| 调试工具    | ChiselDB + TLLog | 无             |

工程化验证的基石

加快设计、测试迭代的方法

## → AMBA CHI 的开源工具生态基本空白

- 在我们决定迁移到 AMBA CHI 的时刻

# 开源互联总线生态——IP

- **互联 = IP + Floorplan**
- **互联结构复杂度不断提高**
  - 各种总线组件的约束、摆放
  - 时钟域、电源域管理
- **SoC 的需求目标多样化**
  - 服务器平台
  - 桌面平台
  - 社区开发、学术研究

**→ IP 本身也构成基础设施的一部分**

# 🚀 开源互联总线生态——IP

| 需求      | TileLink        | AMBA CHI (过去) |
|---------|-----------------|---------------|
| 开箱即用    | 有 (rocket-chip) | 无             |
| 总线组件 IP | 有 (rocket-chip) | 无             |
| 外设组件 IP | 有 (rocket-chip) | 无             |
| 小型互联 IP | 有 (rocket-chip) | 无             |
| 大型互联 IP | 无               | 无             |

} 开源可用性

} 产品级高性能实现

## → AMBA CHI 的开源 IP 生态基本空白

- 在我们决定迁移到 AMBA CHI 的时刻

## 小结 ②

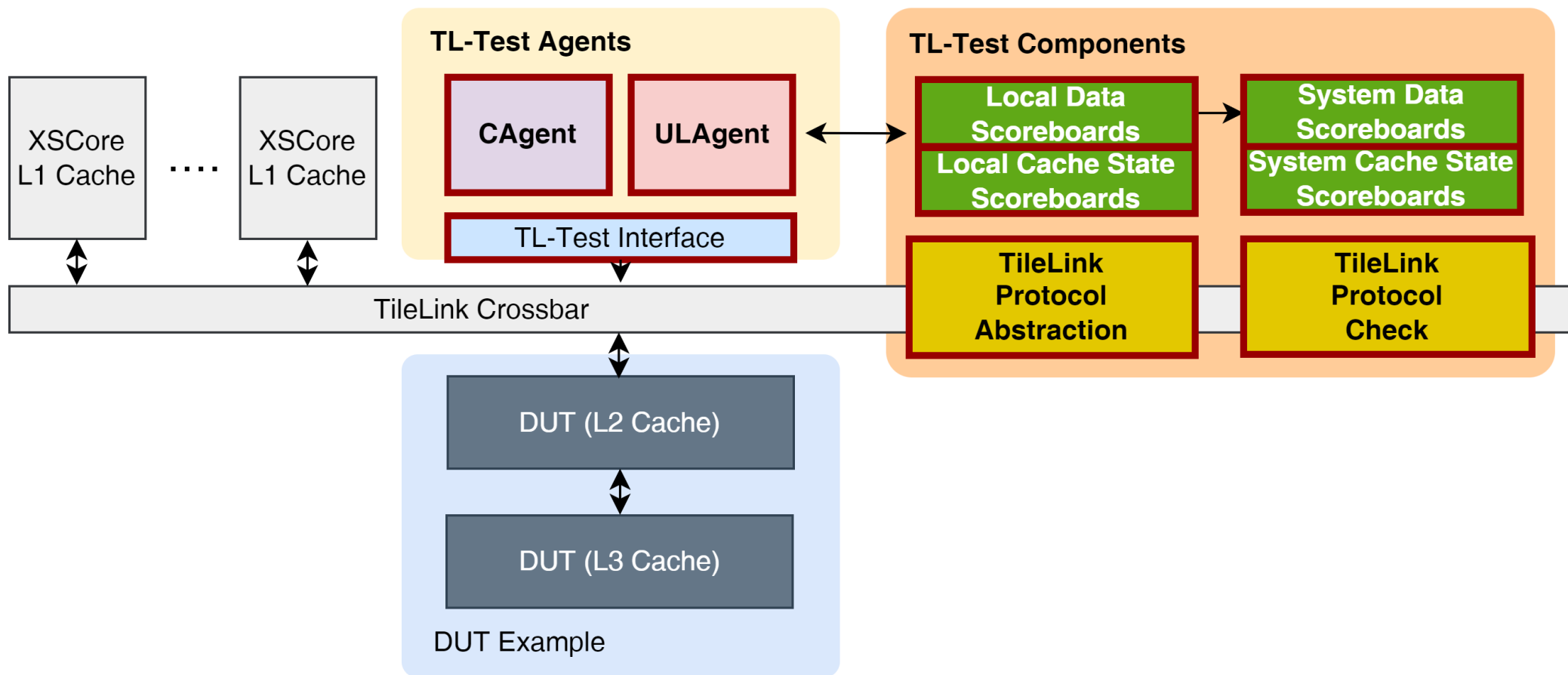
**高扩展、高性能总线开源生态几乎是一片空白**



# 目录

- 背景
- 开源生态
  - 验证方法
  - 硬件 IP
- **解决方案**
  - **过去 (TileLink)**
  - 未来 (AMBA CHI)
- 总结

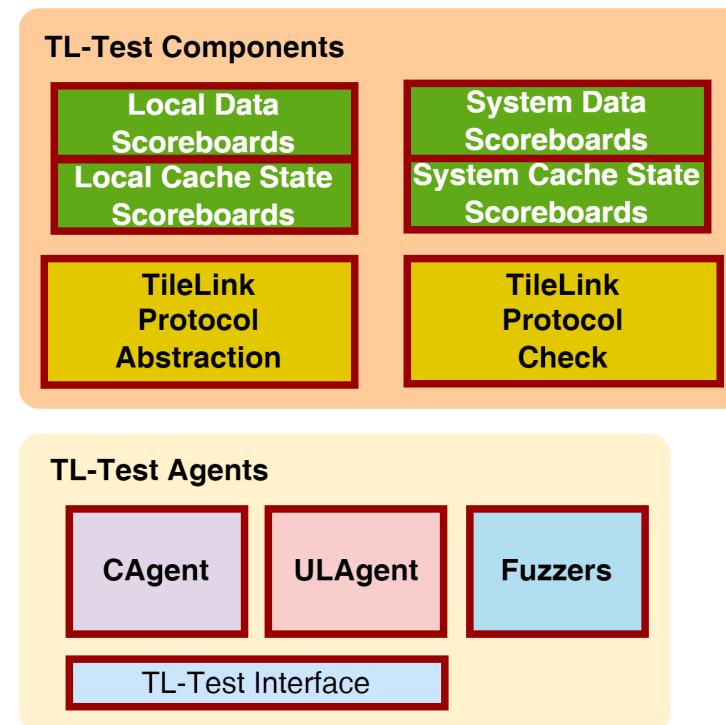
# TL-Test: 基于 TileLink 的多功能缓存验证框架



- 基于 TileLink 总线抽象出的一整套验证、调试基础设施

# 验证基础设施

- **TileLink Protocol Check**
  - 覆盖协议层抽象、事务级抽象
  - 对协议正确性的检查
- **Local & System Scoreboard**
  - 缓存一致性检查
- **CAgent & UAgent**
  - 快速测例构造
- **Fuzzers**
  - 可约束随机测试
  - 在验证前期快速提高测试覆盖率





# 调试辅助工具

## • TLog: 持久化总线事务

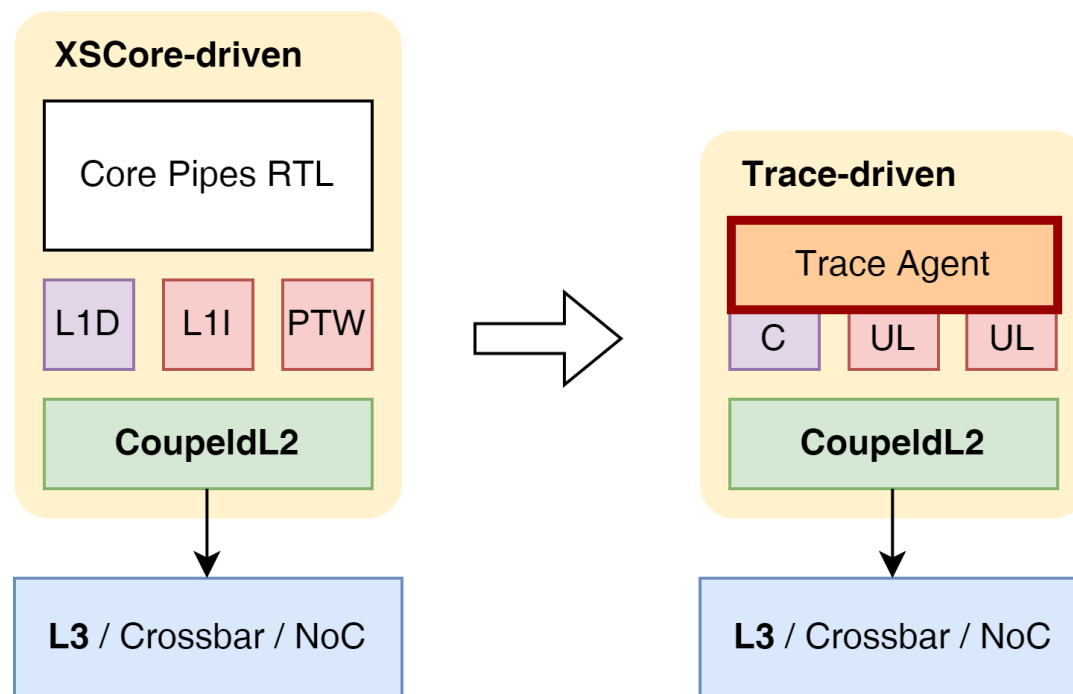
- 基于 ChiselDB (Chisel 下基于 SQLite3 的数据持久化)
- 将运行过程中的总线事务记录到数据库中
- 离线错误检索
- 离线性能分析

| 时间戳       | 监控器名称    | 通道 | Opcode       | 权限升降   | 状态转换信息 | SourceID | SinkID | 地址       | 数据               |                  |                  |                  |
|-----------|----------|----|--------------|--------|--------|----------|--------|----------|------------------|------------------|------------------|------------------|
| 116854134 | L2_L1I_0 | A  | AcquireBlock | Grow   | NtoB   | 1        | 0      | 803d3680 | 0000000000000000 | 0000000000000000 | 0000000000000000 | 0000000000000000 |
| 116854146 | L2_L1I_0 | D  | GrantData    | Cap    | toT    | 1        | 16     | 803d3680 | fa843783f9043903 | 06090063f8f43c23 | f8f4382300093783 | 378300093023c3bd |
| 116854147 | L2_L1I_0 | D  | GrantData    | Cap    | toT    | 1        | 16     | 803d3680 | 03e32e07bb030089 | f0ef8526c881f49b | bb0300893783bbdf | 855a010925832e07 |
| 123191840 | L2_L1I_0 | C  | ReleaseData  | Shrink | TtoN   | 0        | 0      | 803d3680 | fa843783f9043903 | 06090063f8f43c23 | f8f4382300093783 | 378300093023c3bd |
| 123191841 | L2_L1I_0 | B  | Probe        | Cap    | toN    | 0        | 0      | 803d3680 | 0000000000000000 | 0000000000000000 | 0000000000000000 | 0000000000000000 |
| 123191841 | L2_L1I_0 | C  | ReleaseData  | Shrink | TtoN   | 0        | 0      | 803d3680 | 03e32e07bb030089 | f0ef8526c881f49b | bb0300893783bbdf | 855a010925832e07 |
| 123191848 | L2_L1I_0 | D  | ReleaseAck   | Cap    | toT    | 0        | 31     | 803d3680 | 0000000020fab05b | 0000000020fab45b | 0000000020fab85b | 0000000020fab5b  |
| 123191851 | L3_L2_0  | C  | ReleaseData  | Shrink | TtoN   | 31       | 0      | 803d3680 | fa843783f9043903 | 06090063f8f43c23 | f8f4382300093783 | 378300093023c3bd |
| 123191852 | L3_L2_0  | C  | ReleaseData  | Shrink | TtoN   | 31       | 0      | 803d3680 | 03e32e07bb030089 | f0ef8526c881f49b | bb0300893783bbdf | 855a010925832e07 |
| 123191862 | L3_L2_0  | D  | ReleaseAck   | Cap    | toT    | 31       | 16     | 803d3680 | 86a6f7043583dcd5 | f0ef856a865a8762 | 3583b7654481e17f | 865a86a68762f704 |
| 123191863 | L2_L1I_0 | C  | ProbeAck     | Shrink | TtoN   | 0        | 0      | 803d3680 | 0000000000000000 | 0000000000000000 | 0000000000000000 | 0000000000000000 |
| 123191878 | L3_L2_0  | C  | Release      | Report | NtoN   | 30       | 0      | 803d3680 | 5597bf494981aa09 | 8522cc2585930000 | f84ef15dda0f00ef | 00194703b775e84a |

# 🔥 调试辅助工具

## • TLTrace: 基于 trace 的激励生成

- 将处理器核的**访存 trace 转换为测试激励**
- 将 DUT 从处理器核中剥离
- 快速复现功能性问题
- 快速调试性能问题



# TileLink 解决方案

| 需求      | TileLink                            |
|---------|-------------------------------------|
| 协议层抽象   | TL-Test (TileLink Protocol Check)   |
| 事务级抽象   | TL-Test (TileLink Protocol Check)   |
| 一致性检查   | TL-Test (Local & System Scoreboard) |
| 测试环境构造  | TL-Test                             |
| 可约束随机测试 | TL-Test (Fuzzers)                   |
| 快速测例构造  | TL-Test (CAgent & UAgent)           |
| 调试工具    | ChiselDB + TLLog + TLTrace          |

→ 已完成对 TileLink 总线开源生态的需求覆盖



# 目录

- 背景
- 开源生态
  - 验证方法
  - 硬件 IP
- **解决方案**
  - 过去 (TileLink)
  - **未来 (AMBA CHI)**
- 总结

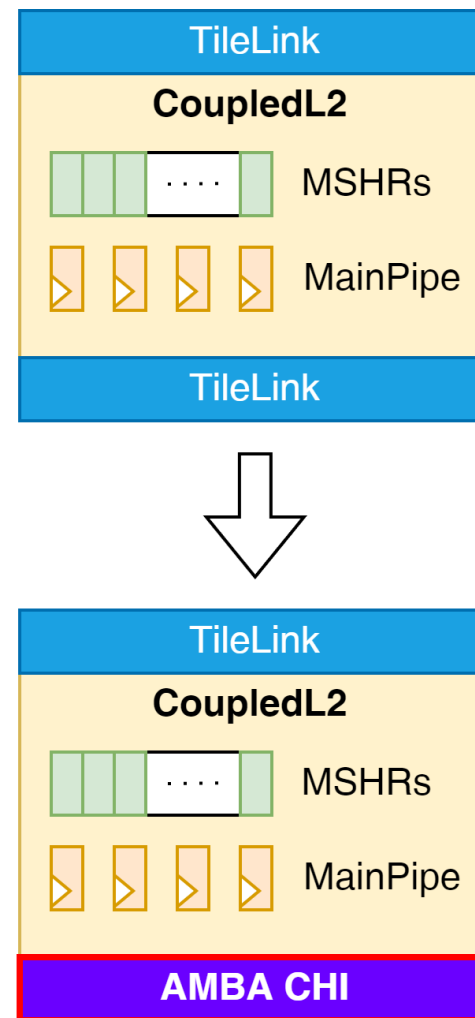
# 硬件架构演进

## • TileLink to CHI

- 一、二级缓存间仍然使用 TileLink 连接
- 二级缓存向下使用 CHI
- **在二级缓存(CoupledL2) 内完成协议转换**
- 测试激励界面依然是 TileLink

## • 实践考虑

- 前期**完全没有** CHI 基础设施支持
- 项目进度限制
  - 很难重写整个缓存子系统
  - 很难同时维护 TileLink 与 CHI 版本的二级缓存



# 🚀 软件工具演进

- TL-Test-new: 支持商业仿真环境的 TL-Test

- 初期没有可用的开源 CHI 下游

- 目前 AMBA CHI 仅有商业 RTL IP 和 VIP

- 保证原有 workflow 可用

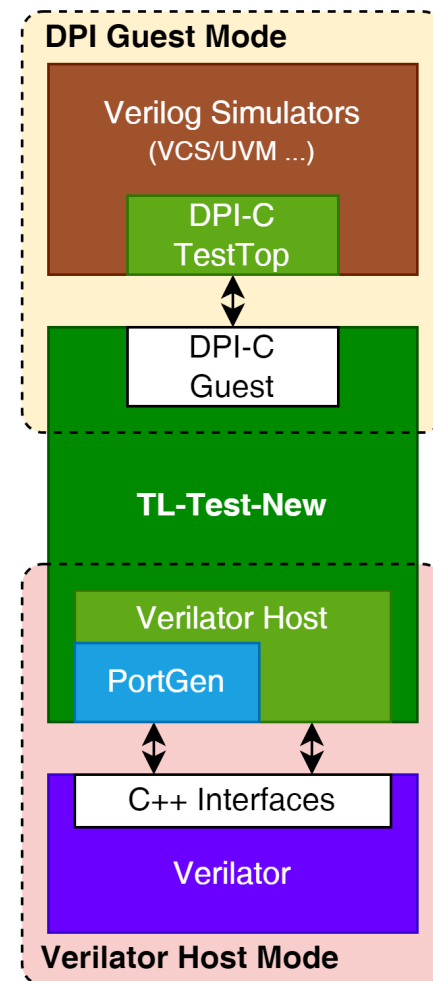
- 完善 ChiselDB 对商业仿真器的支持

- 同时支持开源、商业仿真环境

- 一套代码，部署到多个不同定位的工作流

- 更好的随机激励生成器 (Fuzzer)

→ 开发流程永远需要基础设施的支持



# 软件工具演进

## • CHIron: 开源 AMBA CHI 基础设施

- 用 C/C++ 描述 CHI 协议标准
- 同时支持 CHI Issue B 与 Issue E.b
- 协议层抽象 **已完成**
- 事务层抽象 **开发中**
- 与 ChiselDB 兼容的 CHI Log
  - TileLink 环境下 TLLog 的同定位实现
- **CLog**: ChiselDB 的替代文件格式
  - 多种文件格式定义: CLog.T, CLog.B, CLog.Bz ...
  - 传递 CHI 配置信息
    - 简化不同工作组间的信息传递
  - 相较于 ChiselDB (SQLite3) 达到 **~10x 性能提升** 与 **~0.4x 空间占用**

```
1  $clog.segment.param.begin
2
3  $chi.issue E.b $$
4
5  $chi.width.nodeid      7 $$
6  $chi.width.addr       48 $$
7  $chi.width.rsvdc.req  4 $$
8  $chi.width.rsvdc.dat  4 $$
9  $chi.width.data       256 $$
10 $chi.enable.datacheck 0
11 $chi.enable.poison    0
12 $chi.enable.mpam      0
13
14 $clog.segment.param.end
15
16 $clog.segment.topo.begin
17
18 $comment | Node Id | Node Type |
19 $chi.topo 4      RNF      $$
20
21 $clog.segment.topo.end
22
23 $comment | Time | Node Id | Channel | Flit bits
24 $chi.log 0    4      TXREQ   abcdef0123456789 $end
25 $chi.log 4    4      RXRSP   0123456789abcdef $end
```

# 软件工具演进

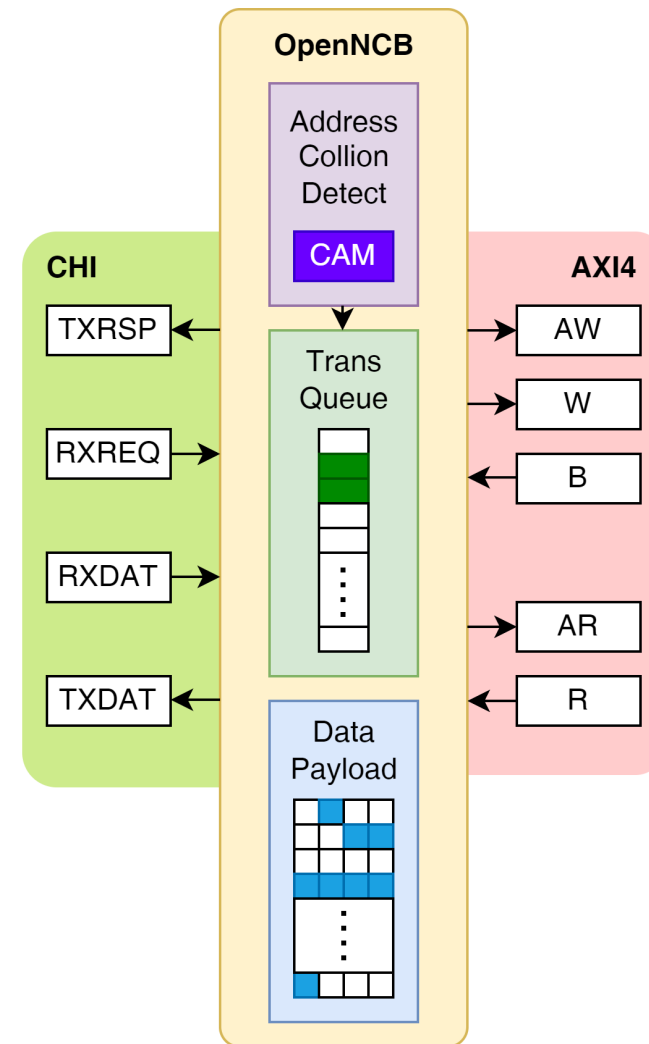
| 需求      | TileLink         | AMBA CHI (过去) | AMBA CHI (未来)                |
|---------|------------------|---------------|------------------------------|
| 协议层抽象   | 有 (TL-Test)      | 无             | 有 (CHIron)                   |
| 链路层抽象   | 不需要              | 无             | 有 (CHIron)                   |
| 事务级抽象   | 有 (TL-Test)      | 无             | 部分 (CHIron)                  |
| 一致性检查   | 有 (TL-Test)      | 无             | 部分,非原生 (TL-Test-new)         |
| 测试环境构造  | 有 (TL-Test)      | 无             | 部分,非原生 (TL-Test-new, CHIron) |
| 可约束随机测试 | 有 (TL-Test)      | 无             | 部分,非原生 (TL-Test-new)         |
| 快速测例构造  | 有 (TL-Test)      | 无             | 部分,非原生 (TL-Test-new)         |
| 调试工具    | ChiselDB + TLLog | 无             | CLog.T + CHILog (CHIron)     |

→ 基础设施仍需要更多努力



# 🌟 开源 CHI SoC IP 实践

- **OpenNCB: 开源高性能 CHI to AXI4 协议桥**
  - 到 AXI4 的非一致性桥接
    - 主要桥接内存设备
  - Credit-controlled Zero Retry
  - **支持参数化配置 outstanding 深度**
  - **支持参数化配置总线数据宽度**
    - CHI: 128, 256, or 512 bits
    - AXI: 32, 64, 128, 256 or 512 bits
  - 可配置的**事务乱序**
    - 全乱序、部分乱序、不乱序
  - 保证可观测性顺序的**事务提前返回**

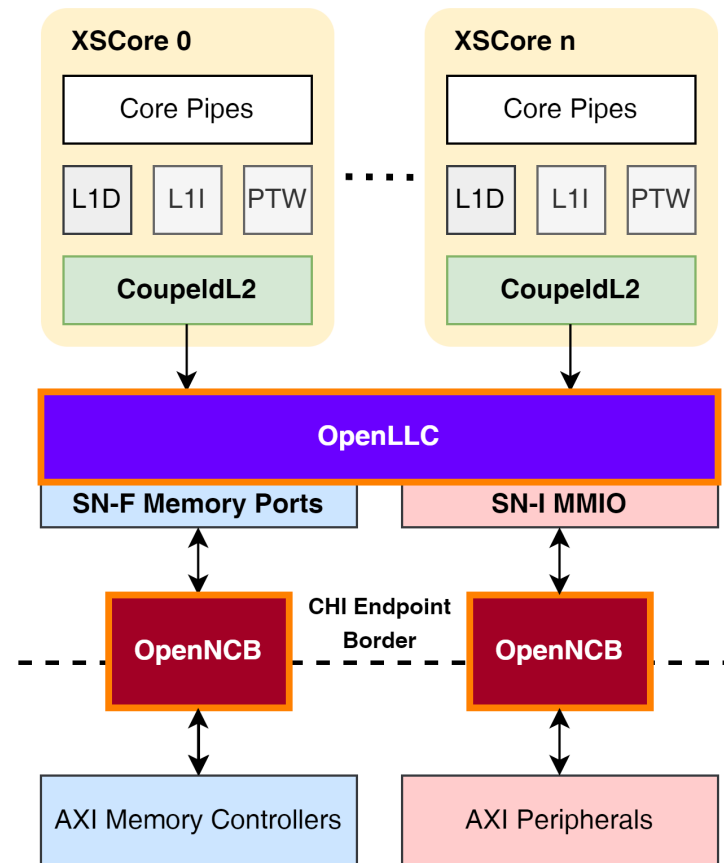


# 🌟 开源 CHI SoC IP 实践

## • OpenLLC: 开源高性能 CHI HN-F 实现

- 向上为 RN-F 端口，向下为 SN-F 端口
- 支持部分必要的 CHI Issue B 事务
- 支持 CMO 请求
- 基于目录的 MESI 一致性协议 (w/ Snoop Filter)
- 支持多种缓存包含策略
  - 单核为 exclusive, 多核为 non-inclusive
- 高并发度设计与非阻塞主流流水线架构
  - 多 slices 并行, 同缓存 set 内请求并行
  - 支持部分事务数据 bypass

→ OpenLLC 与 OpenNCB 组成 CHI 最小互连系统 IP



# 开源 CHI SoC IP 实践

| 需求      | TileLink        | AMBA CHI (过去) | AMBA CHI (未来)           |
|---------|-----------------|---------------|-------------------------|
| 开箱即用    | 有 (rocket-chip) | 无             | OpenLLC + OpenNCB + AXI |
| 总线组件 IP | 有 (rocket-chip) | 无             | OpenLLC, OpenNCB        |
| 外设组件 IP | 有 (rocket-chip) | 无             | OpenNCB + AXI           |
| 小型互联 IP | 有 (rocket-chip) | 无             | OpenLLC + OpenNCB       |
| 大型互联 IP | 无               | 无             | OpenNoC                 |

→ 为开源社区提供更多选择



# 目录

- 背景
- 开源生态
  - 验证方法
  - 硬件 IP
- 解决方案
  - 过去 (TileLink)
  - 未来 (AMBA CHI)
- **总结**

# 总结

- TileLink 无法满足服务器平台产品化需求
- 开源高性能互联总线生态仍有很大空白
- 尝试构建 CHI 的开源生态解决方案

| 项目名称        | 分类     | 定位                       | 状态     |
|-------------|--------|--------------------------|--------|
| TL-Test-new | 工具     | 过渡期的兼容性保障基础设施            | 已完成    |
| CHIron      | 工具     | CHI 的开源基础设施              | 设计、开发中 |
| OpenNCB     | SoC IP | 开源高性能 CHI SN-F to AXI 实现 | 开发中    |
| OpenLLC     | SoC IP | 开源高性能 CHI HN-F 实现        | 开发中    |

- 我们依然在向 CHI 迁移的过渡期
- 开源生态仍需要大家一起来努力

注：1. 以上所有开源项目都以同名仓库的形式存在于 <https://github.com/OpenXiangShan>  
2. 部分未开源就绪的项目会在将来迁移到以上位置

**敬请批评指正!**