Tencent 腾讯

芥子模拟器(risc-v CPU建模)

第一部分 建模的作用

建模的必要性

- 大芯片是复杂系统，单点提升不能生效，需要系统级设计
  1. CPU单纯增加发射路作用很小，甚至有反作用
  2. 单纯加大cache作用很小甚至性能不变
  3. 分支预测采用倒灌方式，性能没有提升
  4. 微架构设计靠多年的经验，很少通过建模做定量分析。软件调试基于FPGA设备或者样片，部分公司依靠vdk精细模型

- 大芯片生态建设
  1. 生态最重要指标是围绕芯片的开发人员数量和质量。开发人员看重编程容易，硬件则看重计算模式的完备性和高效率。
  2. 微架构早期设计阶段就要引入软硬件协同设计：通过建模引入不同程序，测试可能的瓶颈以及解瓶颈
  3. 芯片的编程语言/指令需要定义重要的软硬件界面，确定软件人员的工作：是否需要拼凑数据(SIMD)，是否需要手工处理dma(cache可见)，是否硬件处理各种复杂数据格式转换。

国内IC公司建模的现状

- 第一阶段：无建模
  1. 依靠架构师的个人能力和积累套路
- 第二阶段：soc层级建模：广泛应用在购买关键IP的项目
- 第三阶段：core层级建模，但不能执行指令，有的基于vdk
- 第四阶段：core层级建模，指导架构设计和软硬件协同

建模的作用

- 建模的作用：
  1. 早期的微架构探索
  2. 持续对微架构评估和对齐性能，tradeoff和修改
  3. 在验证阶段，模拟器可作为参考模型辅助验证，快速定位逻辑设计错误
  4. 系统软件的开发和定位
  5. 硅后验证和测试用例

- 建模需要达到的目标
  1. 基于CA对齐（cycle accurate），复杂硬件要简化（ddr/pcie）
  2. 基于电路设计原则抽象软件代码
  3. 可直接执行程序和指令，而非cost分析
  4. 快速的开发语言和电路基础
  5. 性能模型和功能模型分离，适配软件的硅后模型分离

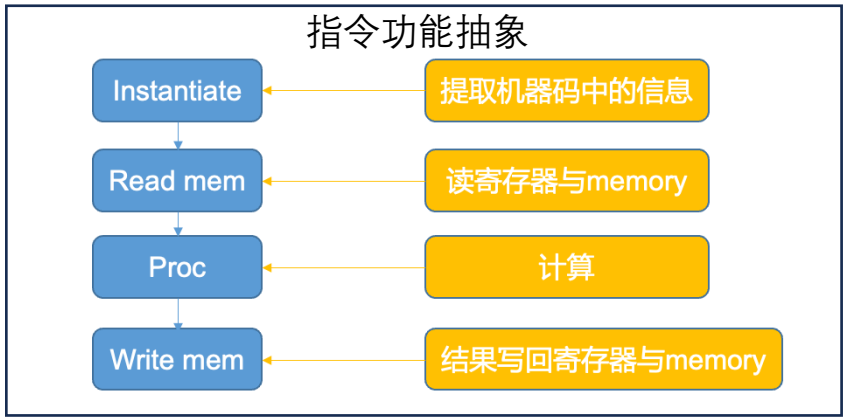
第二部分 cpu建模的具体实践

芥子Riscv CPU模拟器

- 芥子模拟器是腾讯芯片团队基于GEM5开源模拟器开发的Riscv CPU模拟器。
  https://github.com/OpenXiangShan/PenglaiJiezi
- 开源版本：specint 12.2/Ghz，specfp 15.3/Ghz
- 芥子模拟器开源版本首先与香山南湖CPU微架构进行了架构对齐，并在其基础上进行了如下微架构优化：
  - CPU前端： 实现了增强版本的SC预测器, Loop预测器, 和基于推测链表栈和提交栈的增强型RAS, 以及Icache性能相关的优化。
  - CPU后端：对LSQ、ROB等乱序组件的配置进行分析和优化。实施了混合 RMAP 和 HBMAP 解决方案以增强重命名表恢复解决方案。
  - CPU Memory子系统：实现了 Bingo 和 SPP 预取器，它们位于混合缓存级别并预取到当前或低级缓存。

功能与性能解耦的处理器建模方法

- 兼顾功能&性能的准确性和工程的易实现性
- 功能模拟：
  - 功能准确：
    - 与芯片执行结果**bit级一致**
  - 开发&执行快速：
    - 以**指令粒度**模拟处理器功能
    - 指令功能抽象为**4个函数接口**
- 性能模拟：
  - 性能准确：
    - 与芯片执行结果**cycle级一致**
  - 开发&执行快速：
    - Event机制模拟电路cycle行为
    - Proc模拟**反压点，仲裁点，产生请求的节点**
    - 在流水线中**复用指令功能函数**

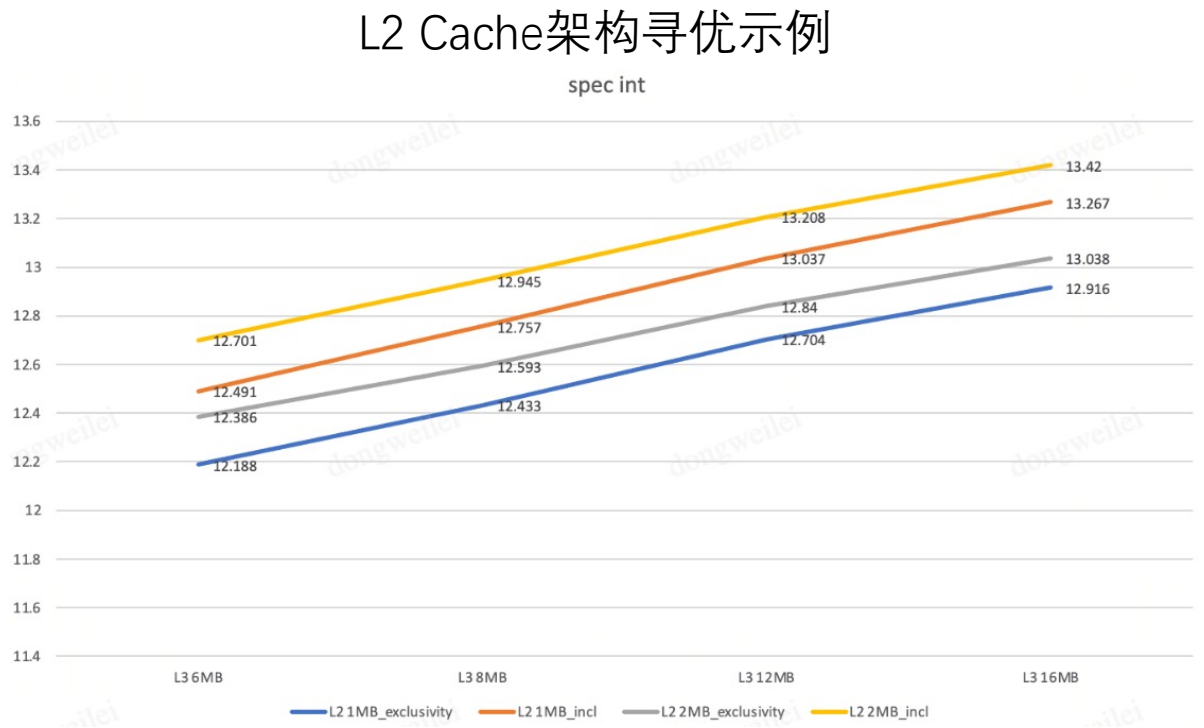


Event + Proc完成对电路抽象。

- Cycle0: Proc0 模拟组合逻辑功能，将计算结果进行缓存，并知会调度器在Cycle1触发Event1，Event1对应Proc1的执行。
- Cycle1: 调度器触发Event1，Proc1开始执行，从reg0获得cycle1对应的寄存器值，并运算。

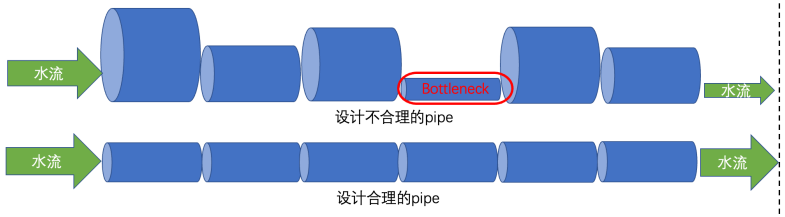
基于设计空间的Profiling架构探索

- 设计空间抽象：将硬件模块的架构设计空间抽象为参数，通过参数调节来对架构进行寻优。
  - L2 Cache参数：
    - size，
    - associaty，
    - clusivity，
    - mshrs，
    - replacement_polity，
    - directory structure，
    - Data Storage structure.

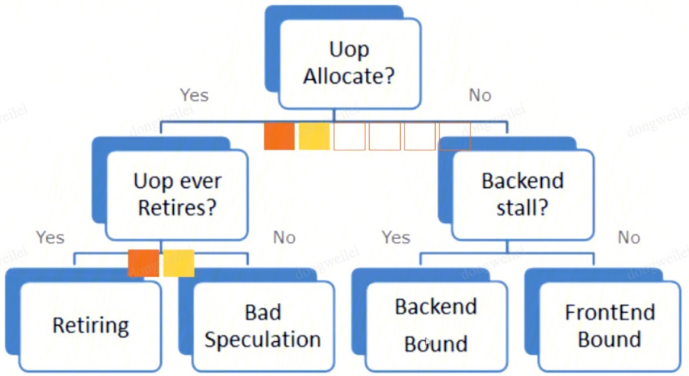


基于Top Down瓶颈分析的处理器架构探索

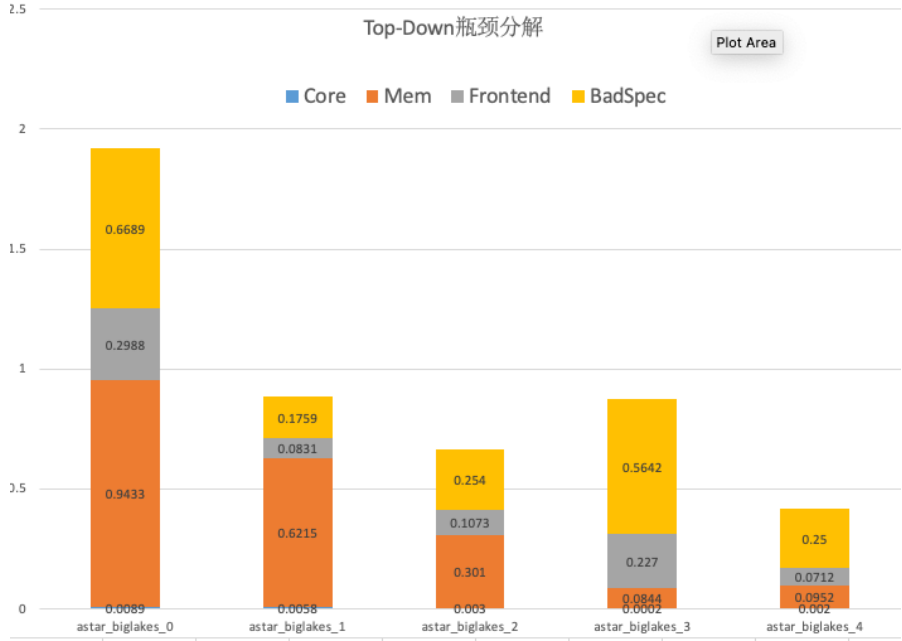
- 处理器的流水线可以抽象为管道，整体性能受限于Bottleneck节点。



- 从CPU的dispatch流水线级的视角进行层层分解，量化各级流水线的Bottleneck占比。



- 输出各个Benchmark场景的瓶颈量化分析结果。



模型与硬件实现方案联动，保证电路的可实现性

微架构资源变化实时反映在面积与性能对比图上。快速进行架构决策。

指数计算指令硬件设计方案

硬件设计方案一：指数计算资源复用1拍，两条指令间间隔0拍

scalar hareware slot	instruction	dtype	read port	write port	EX1	EX2	EX3	EX4	EX5	EX6	EX7	EX8	EX9	EX10
	VRSQRT	f32 f16	R2 PR1	W1	1fa	1fb	1fc	1fd	1c	1fe	1ff	1fh	1fi	1fj
	VEXP	f32 f16	R2 PR1	W1	1fa	1fb	1fc	1fd	1c	1fe	1ff	1fh	1fi	1fj

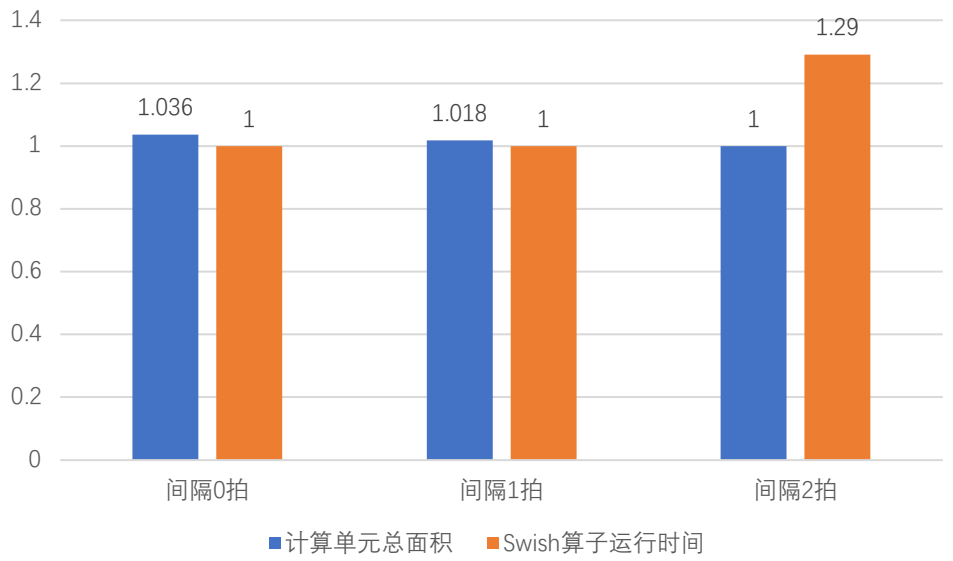
硬件设计方案二：指数计算资源复用2拍，两条指令间间隔1拍

instruction	dtype	read port	write port	EX1	EX2	EX3	EX4	EX5	EX6	EX7	EX8	EX9
VEXP	f32	R2 PR1	W1	1fa	1fa	z	z	1d0	1d0	lut_out	lut_out	z
VEXP	f16	R2 PR1	W1	1fa	1fa	z	z	1d0	1d0	lut_out	lut_out	z

硬件设计方案三：指数计算资源复用4拍，两条指令间间隔3拍

instruction	dtype	read port	write port	EX1	EX2	EX3	EX4	EX5	EX6	EX7	EX8	EX9	EX10	EX11
VEXP	f32	R2 PR1	W1	1fa	1fa	z	z	1d	1d	lut_out	lut_out	z		
VEXP	f16	R2 PR1	W1	1fa	1fa	1fa	1fa	1d	1d	1d	1d	lut_out	lut_out	z

指数计算指令模型评估结果



	间隔0拍	间隔1拍	间隔2拍
计算单元总面积变化	1.036	1.018	1
Swish算子运行时间	1	1	1.29