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

香山开源处理器社区 XiangShan Open-Source Processor Community



香山昆明湖
后端流水线的设计演进

曾锦鸿[1] 张紫飞[2] 胡轩[1] 唐浩晋[1]刘泽昊[1]肖飞豹[2] 程光辉[2] 游朝阳[2] 林志达[1]

吉骏雄[1] 朱夏楠[1] 王桑平[4] 赵祖康[4] 汤芮[1] 胡格致[3] 白冰洋[5] Marc Macros[6] 王欧宇[1]

[1]中国科学院计算技术研究所 [2]北京开源芯片研究院

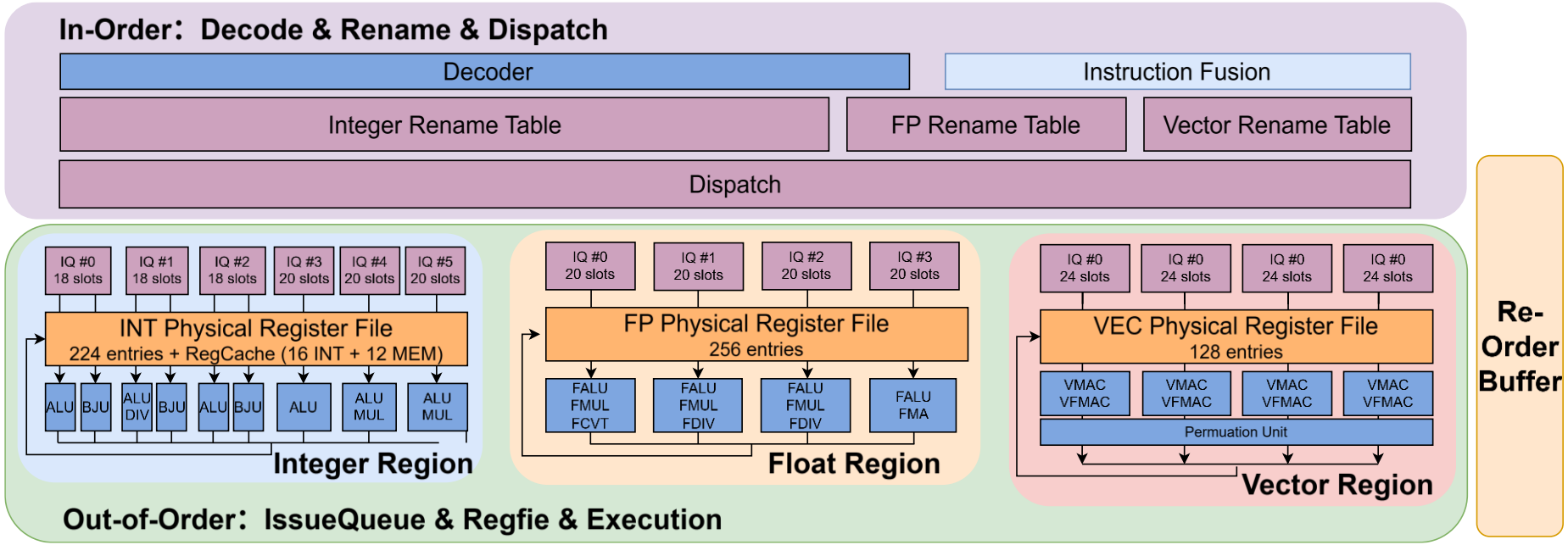
[3]中国科学院大学 [4]上海科技大学 [5]深圳技术大学 [6]巴塞罗那超级计算中心

2026 年 10 月 18 日@第五届 RISC-V 中国峰会

昆明湖 V3 后端架构总览

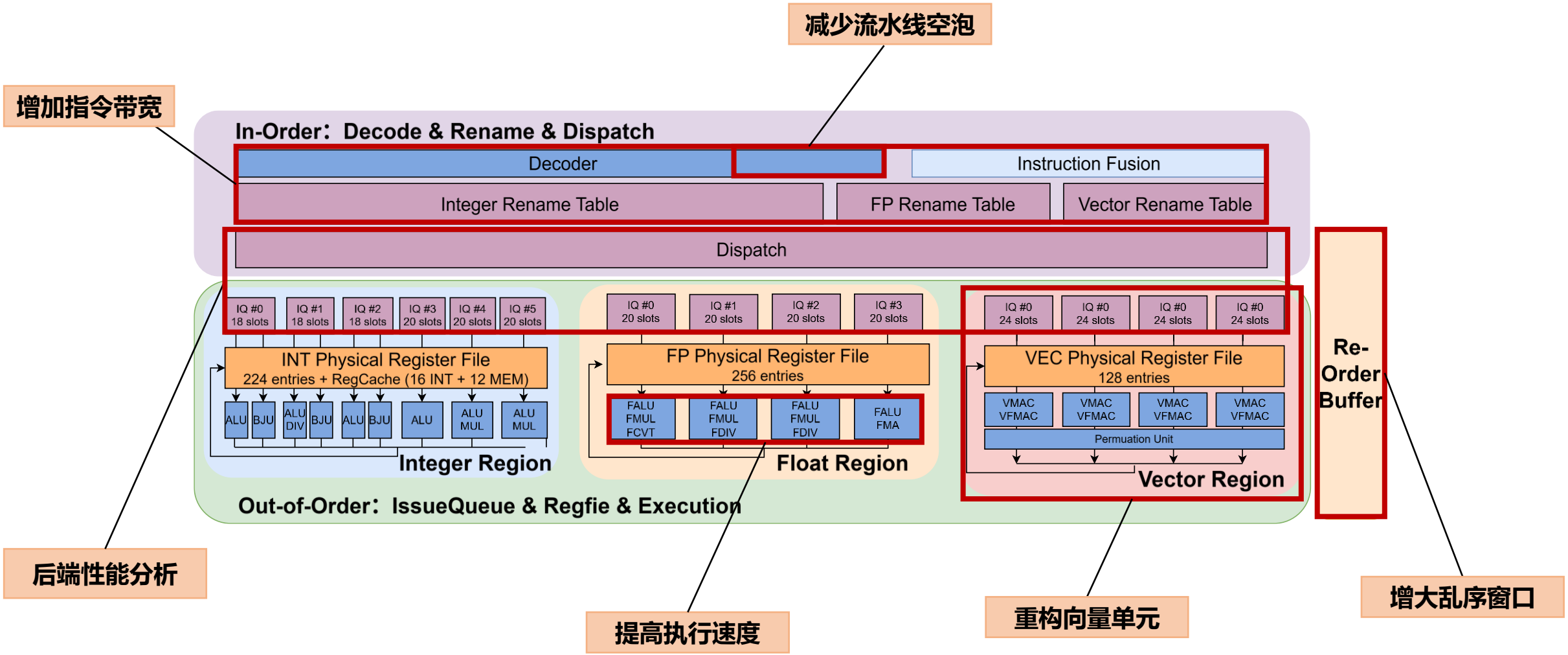
① 顺序阶段

② 乱序阶段



③ 重定序缓冲

后端架构优化方向

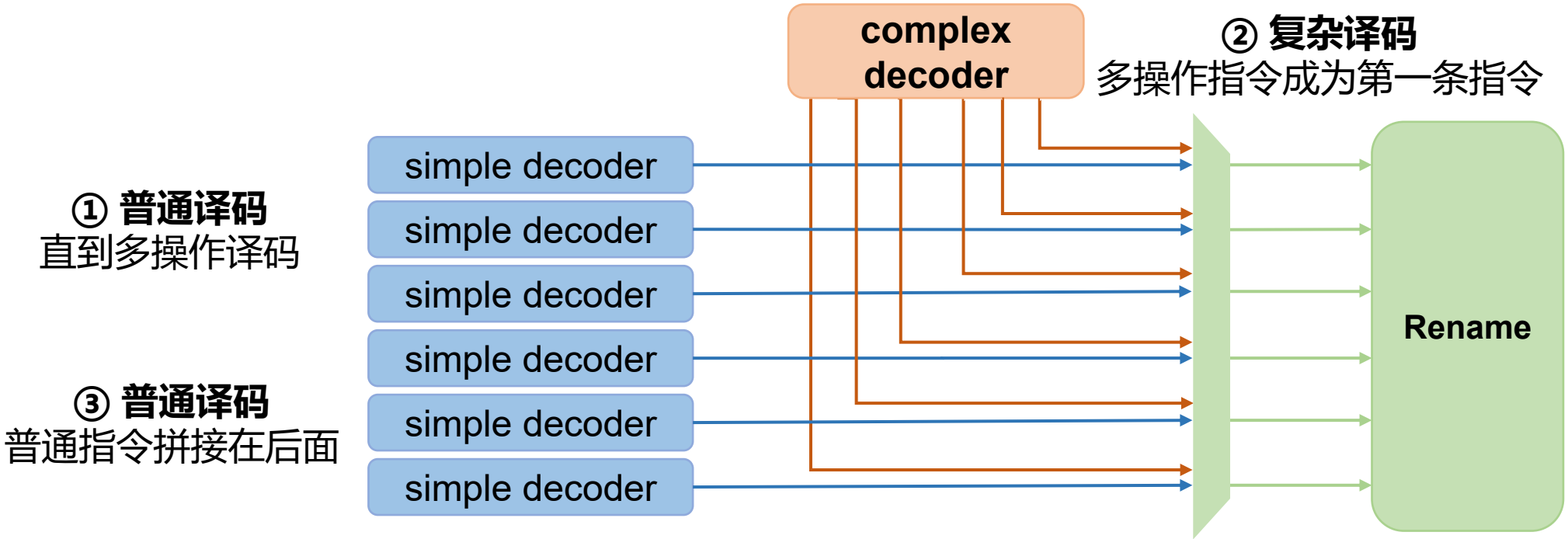

减少流水线空泡
增加指令带宽
In-Order：Decode & Rename & Dispatch
Decoder
Instruction Fusion
Integer Rename Table
FP Rename Table
Vector Rename Table
Dispatch
IQ #0 18 slots
IQ #1 18 slots
IQ #2 18 slots
IQ #3 20 slots
IQ #4 20 slots
IQ #5 20 slots
INT Physical Register File
224 entries + RegCache (16 INT + 12 MEM)
ALU
BJU
ALU DIV
BJU
ALU
BJU
ALU
ALU MUL
ALU MUL
Integer Region
IQ #0 20 slots
IQ #1 20 slots
IQ #2 20 slots
IQ #3 20 slots
FP Physical Register File
256 entries
FALU FMUL FCVT
FALU FMUL FDIV
FALU FMUL FDIV
FALU FMA
Float Region
IQ #0 24 slots
IQ #0 24 slots
IQ #0 24 slots
IQ #0 24 slots
VEC Physical Register File
128 entries
VMAC VFMAC
VMAC VFMAC
VMAC VFMAC
VMAC VFMAC
Permuation Unit
Vector Region
Out-of-Order：IssueQueue & Regfie & Execution
Re-Order Buffer
后端性能分析
提高执行速度
重构向量单元
增大乱序窗口

参数对比：昆明湖 V1 → V2 → V3

模块和队列	V1	V2	V3
Decode Width	6	6	8
Rename Width	6	6	8
Commit Width	6	6	8
Dispatch latency	2	1	1
Regfile - Integer	192	224	224
Regfile - Float	192	192	256
Regfile - Vector	192 (share with float)	128	128
RoB Size	256	160*6 (compress)	160*8 (compress)
ALU / BJU / MUL	4/2/2	4/3/2	6/3/2
FALU / FMUL / FDIV	4/4/2	3/3/2	4/4/1
VALU / VFALU	-	2/2	4/4
AGU LD/ST	3/3	3/2	3/2

指令译码 – 消除多操作空泡

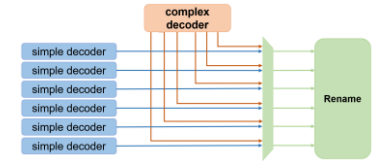
- 旧版实现包含两种译码器
  - Simple Decoder：普通指令译码器，每条译码通道各一个
  - Complex Decoder：复杂指令译码器，译码第一条指令



举例：旧译码方案产生空泡

待译码指令序列，译码宽度 6

Inst1–2	各 1 uop
Inst3	**8 uops**
Inst4-5	各 1 uop
Inst6	**4 uops**
Inst7-8	各 1 uop



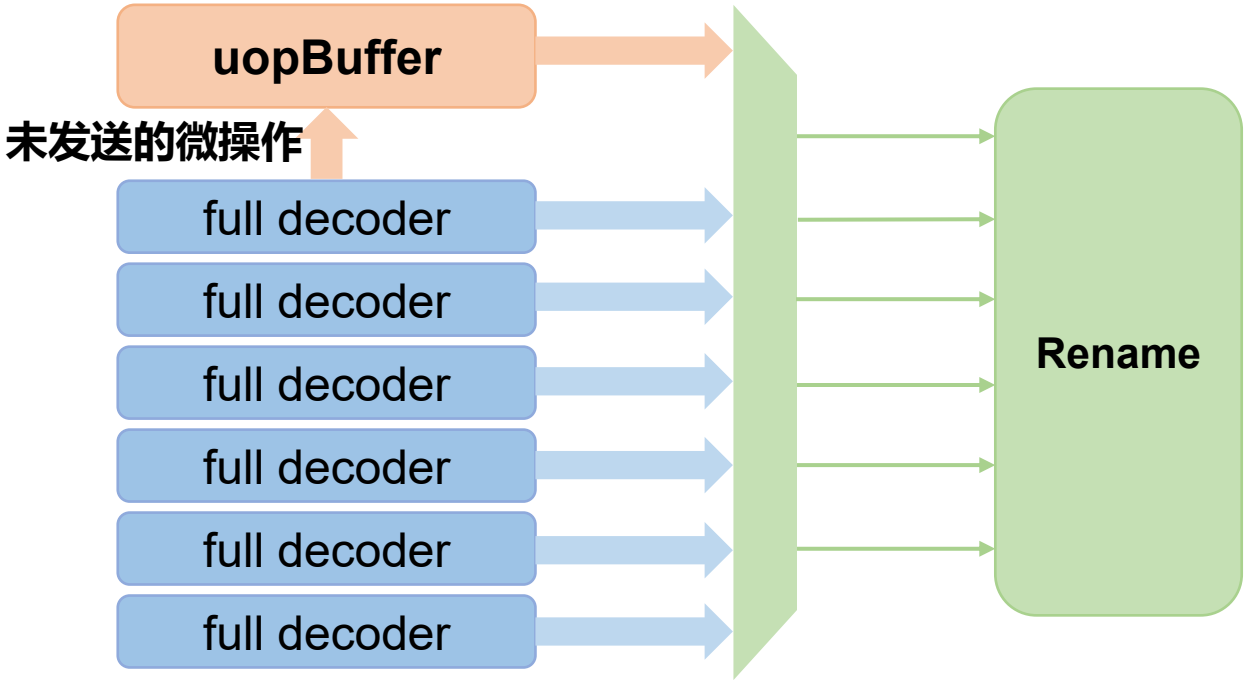
指令拆分导致空泡

T	T+1	T+2	T+3
uop1.1	uop3.1	uop3.7	uop6.1
uop2.1	uop3.2	uop3.8	uop6.2
Bubble	uop3.3	uop4.1	uop6.3
Bubble	uop3.4	uop5.1	uop6.4
Bubble	uop3.5	**Bubble**	uop7.1
Bubble	uop3.6	**Bubble**	uop8.1

4 周期译码 18 个 uop，6 个 bubble

无空泡译码

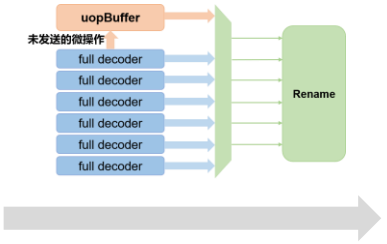
- 译码通道：同时译码普通指令和复杂指令
- uopBuffer：暂存已译码但是未全部送出的微操作



举例：无空泡译码充分利用指令带宽

待译码指令序列，译码宽度 6

Inst1–2	各 1 uop
Inst3	**8 uops**
Inst4-5	各 1 uop
Inst6	**4 uops**
Inst7-8	各 1 uop



T	T+1	T+2
uop1.1	uop3.5	uop6.1
uop2.1	uop3.6	uop6.2
uop3.1	uop3.7	uop6.3
uop3.2	uop3.8	uop6.4
uop3.3	uop4.1	uop7.1
uop3.4	uop5.1	uop8.1

3 周期译码 18 个 uop，0 个 bubble

增加窗口 – 提升ROB的有效容量

- 旧版简单压缩：按指令类型压缩 ROB 表项

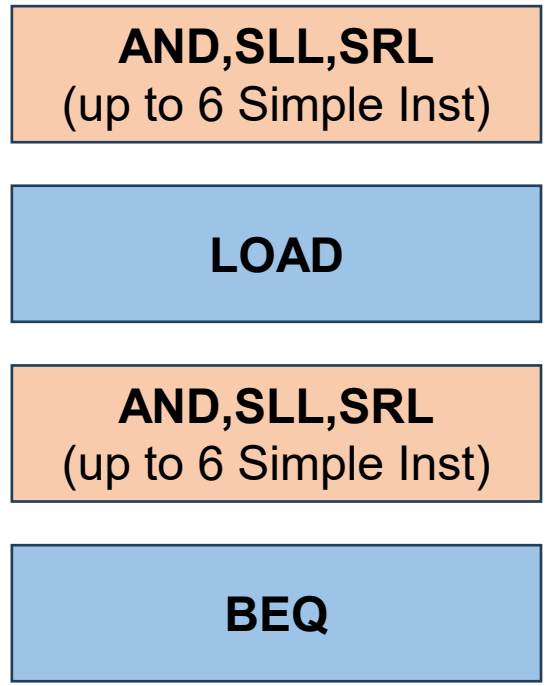
简单指令 **连续简单指令共享 ROB 表项**

不刷新流水线，如运算指令

复杂指令 **每条指令独占 ROB 表项**

可能刷新流水线的指令，如访存、分支指令

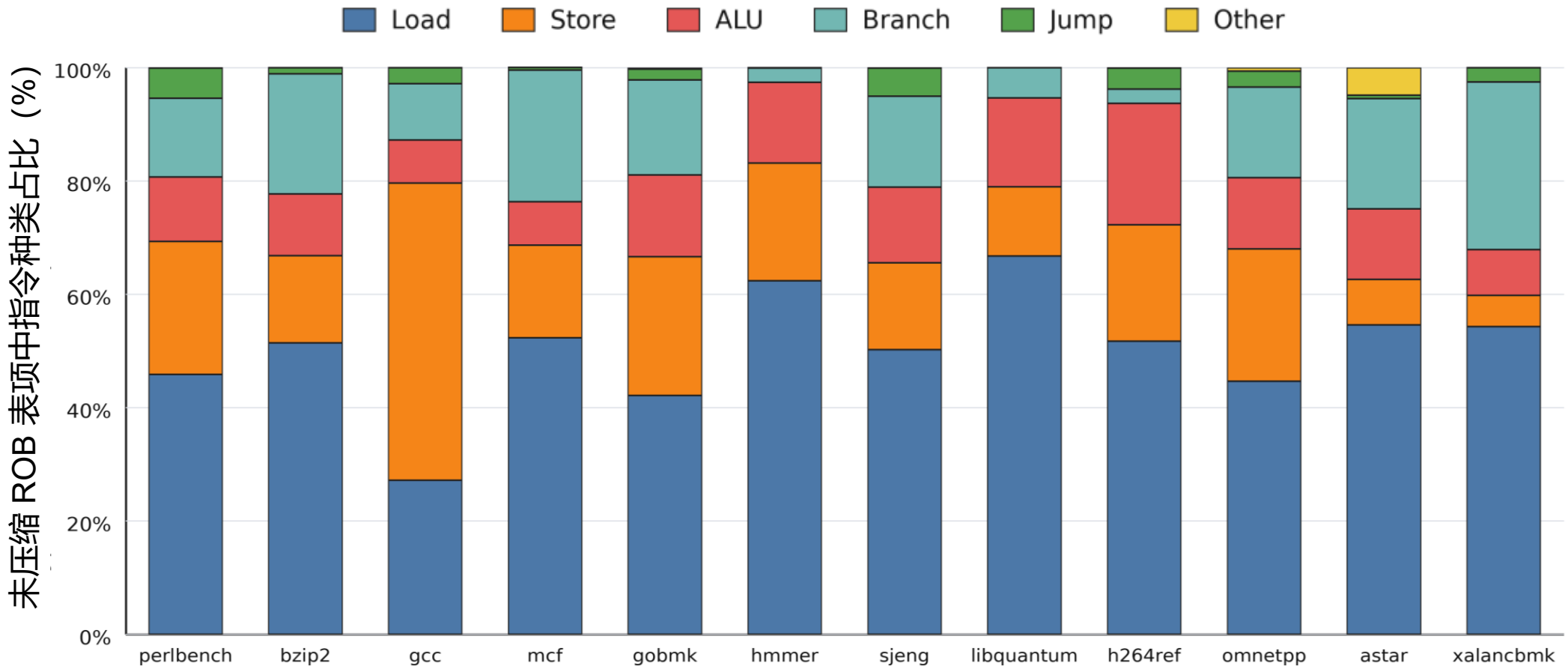
- 压缩效果差
  - 访存、分支指令频繁，打断简单指令的连续
  - 平均压缩率仅 **1.4x**



简单压缩后的ROB

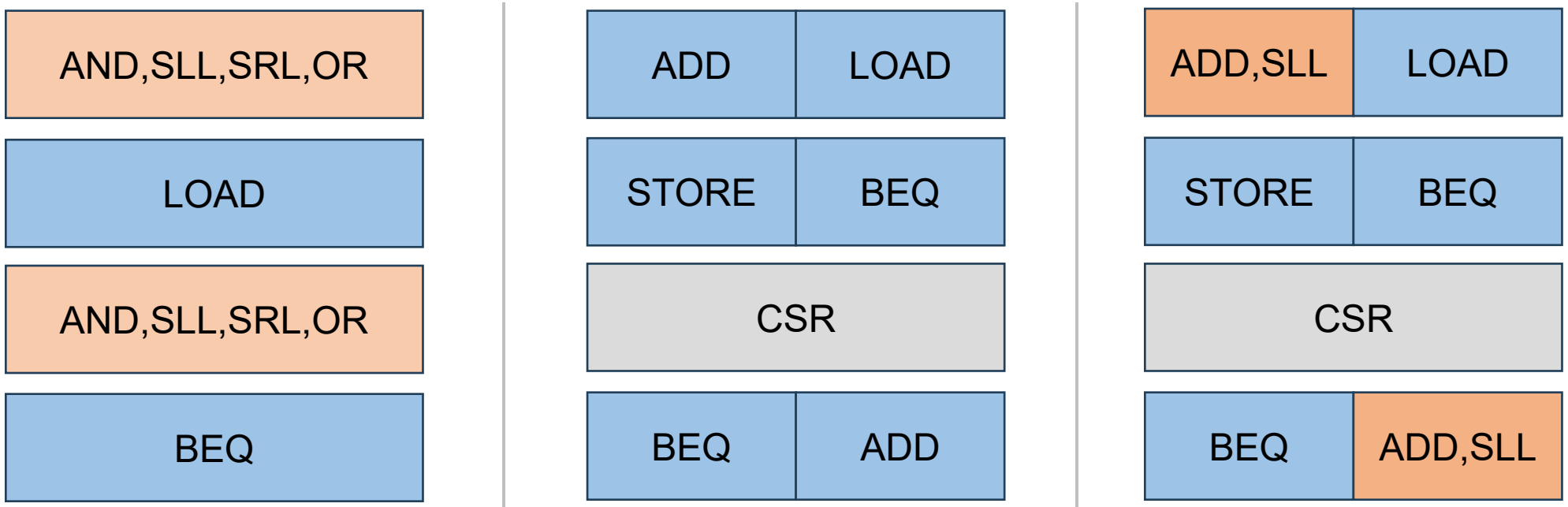
每个方框表示 1 个ROB表项

未压缩 ROB 表项的指令类型构成



访存指令和控制流指令是未压缩的主要来源

混合ROB压缩方案-分析



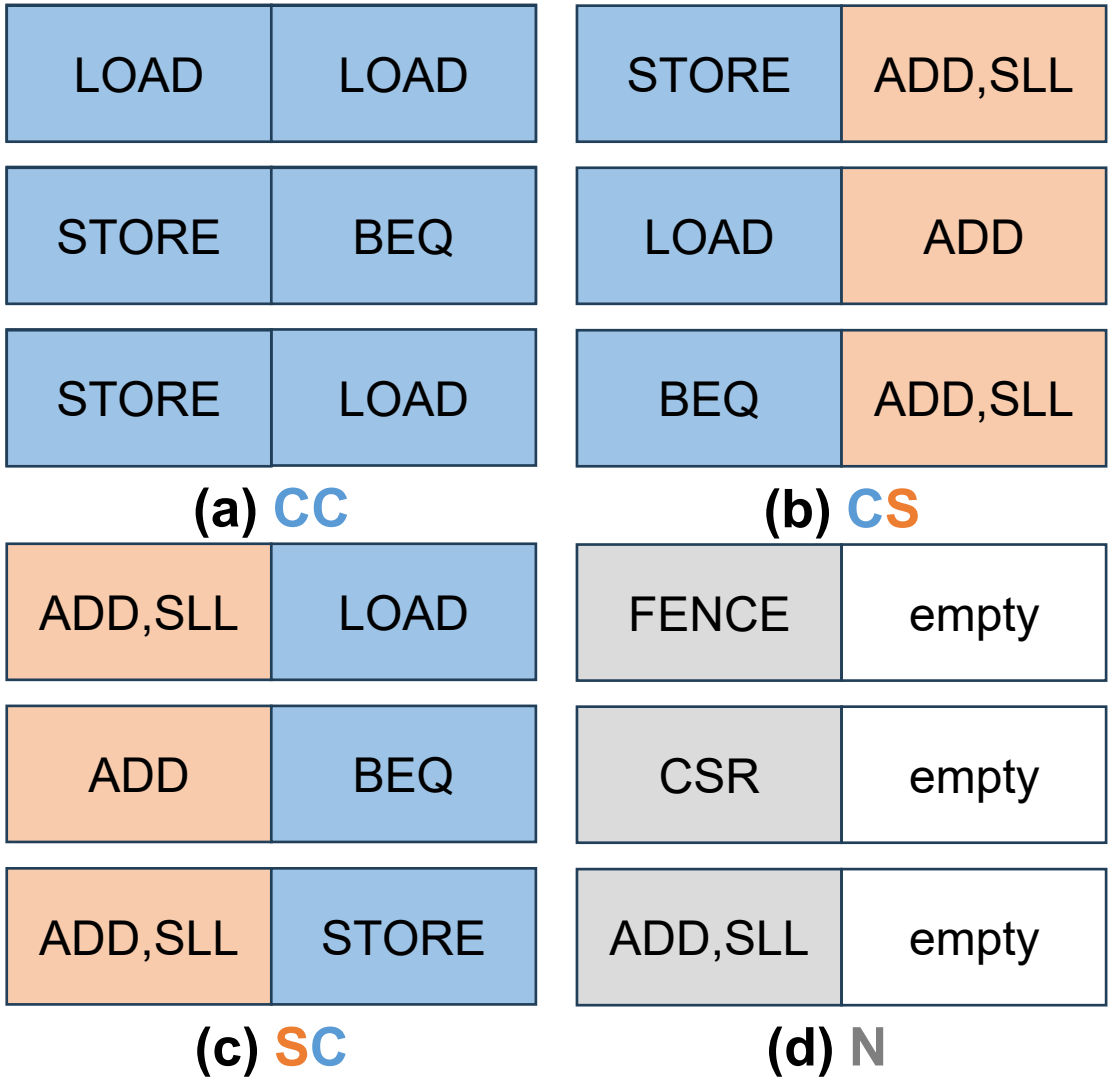
简单压缩 **双指令压缩** **混合压缩**

简单压缩： 压缩连续的简单指令到一个 ROB 项
双指令压缩： 压缩连续的最多两条复杂指令
混合压缩： 同时支持简单指令的高密度压缩，和复杂指令的压缩

混合压缩 – 压缩方法

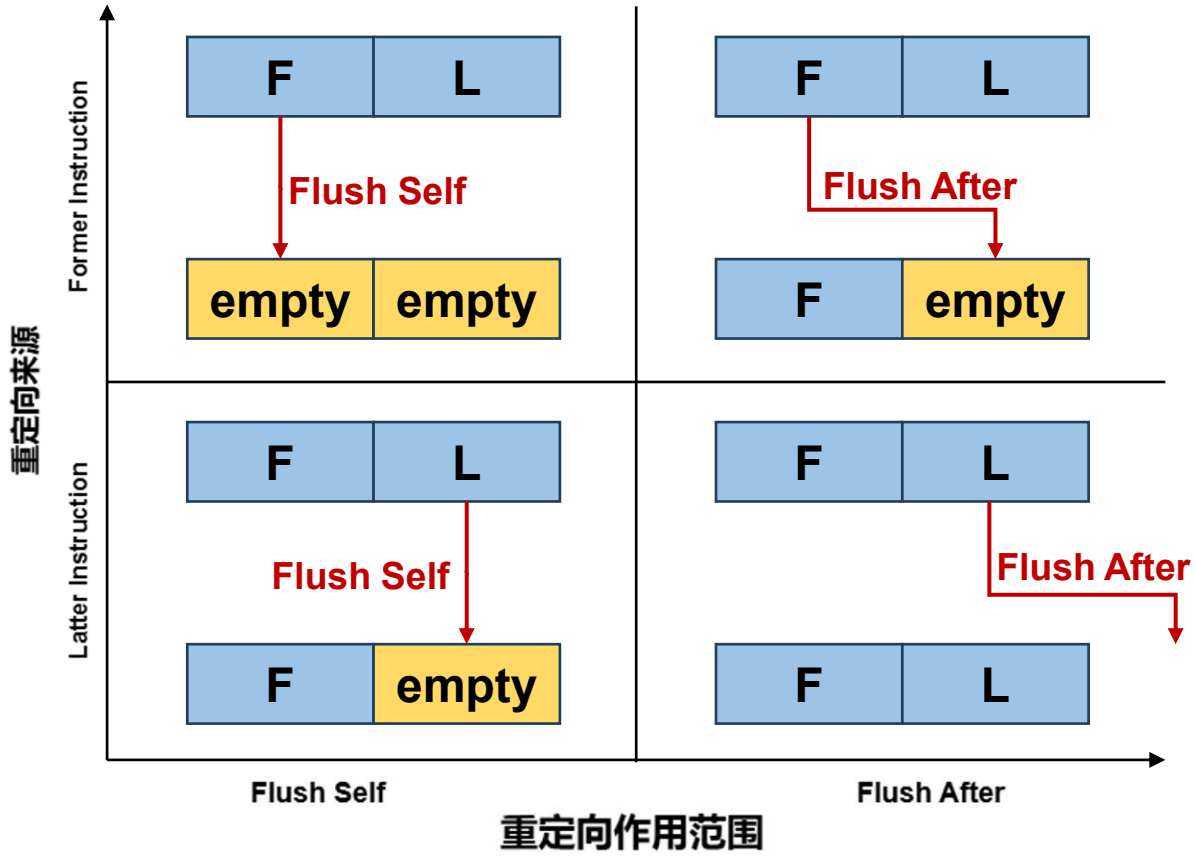
- **指令分类**
  - 简单指令 **C**： 整数和浮点运算类
  - 复杂指令 **S**：访存、控制流类等
  - 不可压缩 **N**：原子、CSR类等

- **四种组合**
  - **C**：压缩表项对应槽为复杂指令
  - **S**：压缩表项对应槽为简单指令组
  - **N**：不可压缩项

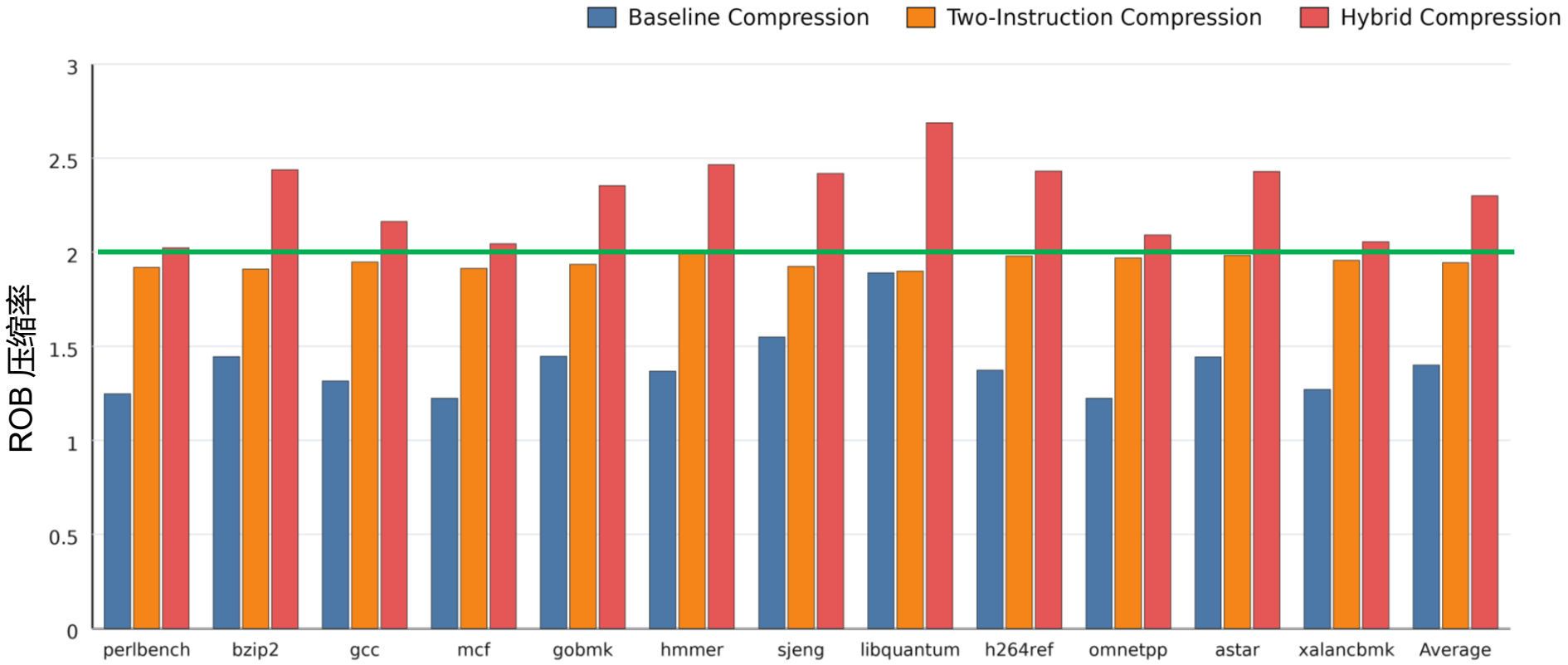


混合压缩 – 重定向

- 同一 ROB 表项内可能存在两个重定向触发源（Former, Latter）
- **重定向作用范围**
  - **Flush Self**：Load 重发、异常
  - **Flush After**：分支误预测、Fence
- **重定向后 ROB 表项为**
  - 无效状态
  - SC、CC
  - NORMAL

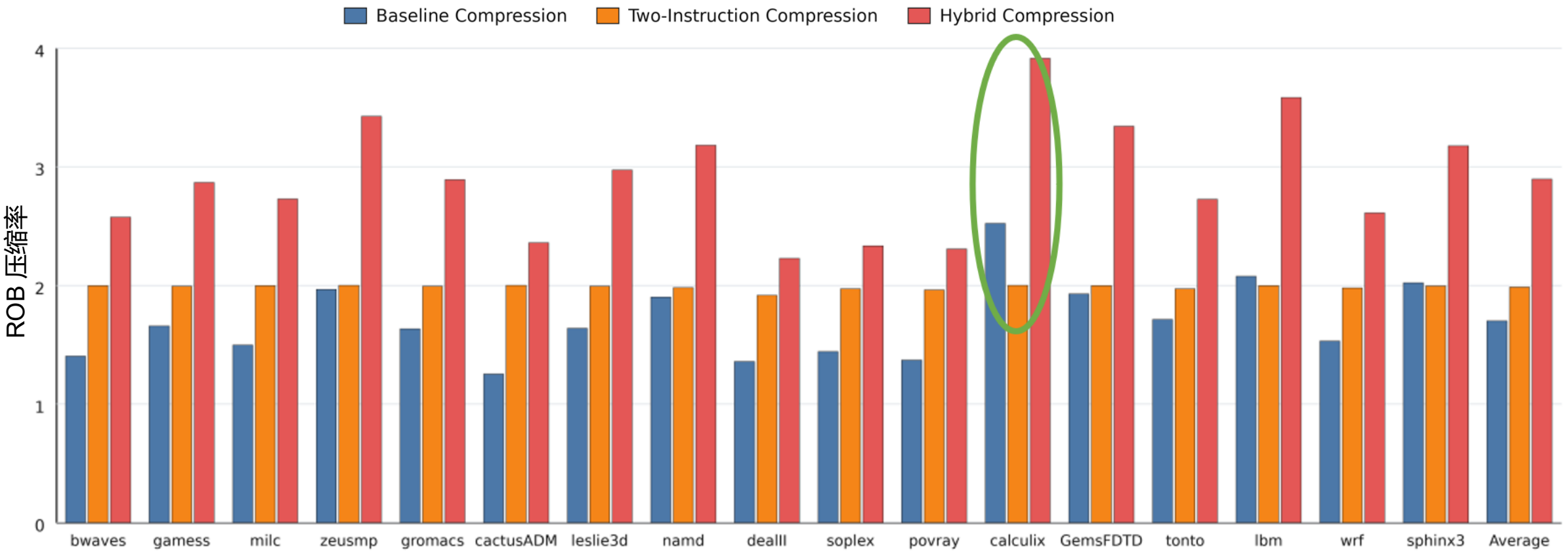


ROB 压缩率对比（SPECint2006）



- V2 压缩的平均 ROB 压缩率为 1.4x
- 双指令压缩的 ROB 压缩率在所有负载上均接近（不超过） 2x
- V3 压缩的平均 ROB 压缩率为 2.3x

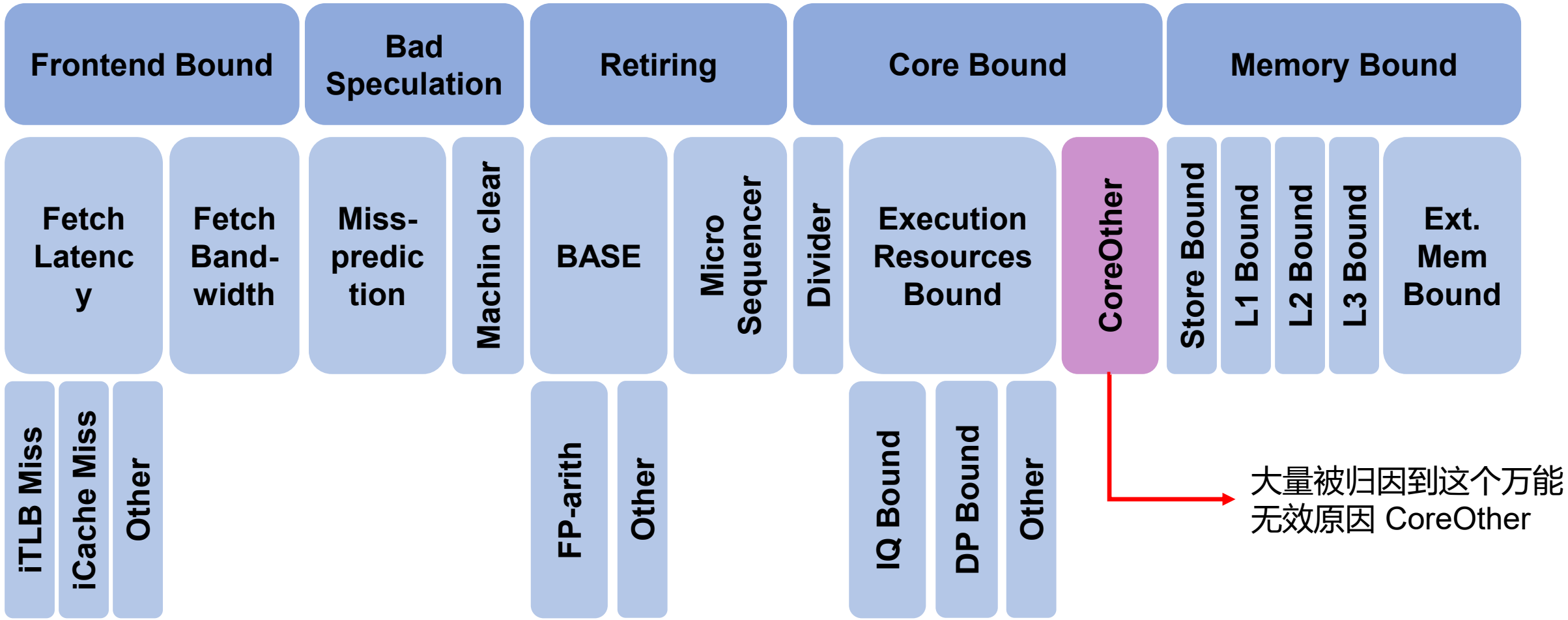
ROB 压缩率对比（SPECfp2006）



- V3 压缩的平均 ROB 压缩率为 2.9x
- 双指令压缩在绝大多数负载上优于 V2 压缩
- V3 压缩在所有负载上都能获得更高的压缩率

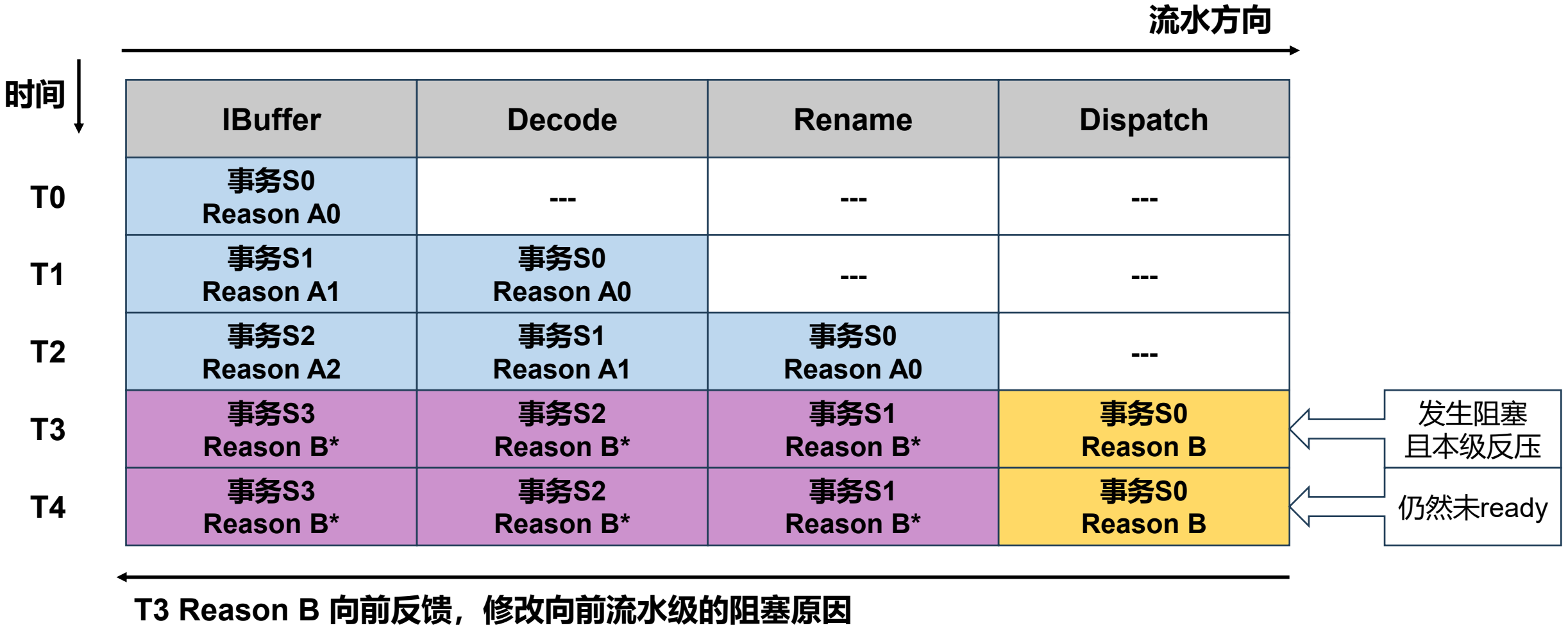
TopDown 性能分析

- 自顶向下地将处理器执行过程中损失的吞吐归因到对应瓶颈类别



旧 TopDown 存在的问题

- 后流水级停顿导致 reason 覆写错配



旧 TopDown 存在的问题

- 部分后端的 stall/bubble 无法被完整覆盖，最终表现为“未知归因”

流水方向 →

时间 ↓

	IBuffer	Decode	Rename	Dispatch
T0	事务S0 Reason A0	---	---	---
T1	事务S1 Reason A1	事务S0 Reason A0	---	---
T2	事务S2 Reason A2	事务S1 Reason A1	事务S0 Reason A0	---
T3	事务S3 Reason A3	事务S2 Reason A2	事务S1 Reason A1	事务S0 Reason UNKNOW
T4	事务S4 Reason A4	事务S3 Reason A3	事务S2 Reason A2	事务S0 + 事务S1 Reason UNKNOW

IQ 部分阻塞

改进策略：Reason 随流水位传递

- **核心变化**

全局旁路 Reason → **Reason 与流水位绑定**

- **归因流程**

01 本级判定

各级独立判断当前槽位为何没有有效推进

只生成本级的 reason

02 绑定传递

Reason 作为槽位属性，随 Stall 和 Bubble 逐级流动

不再使用全局旁路

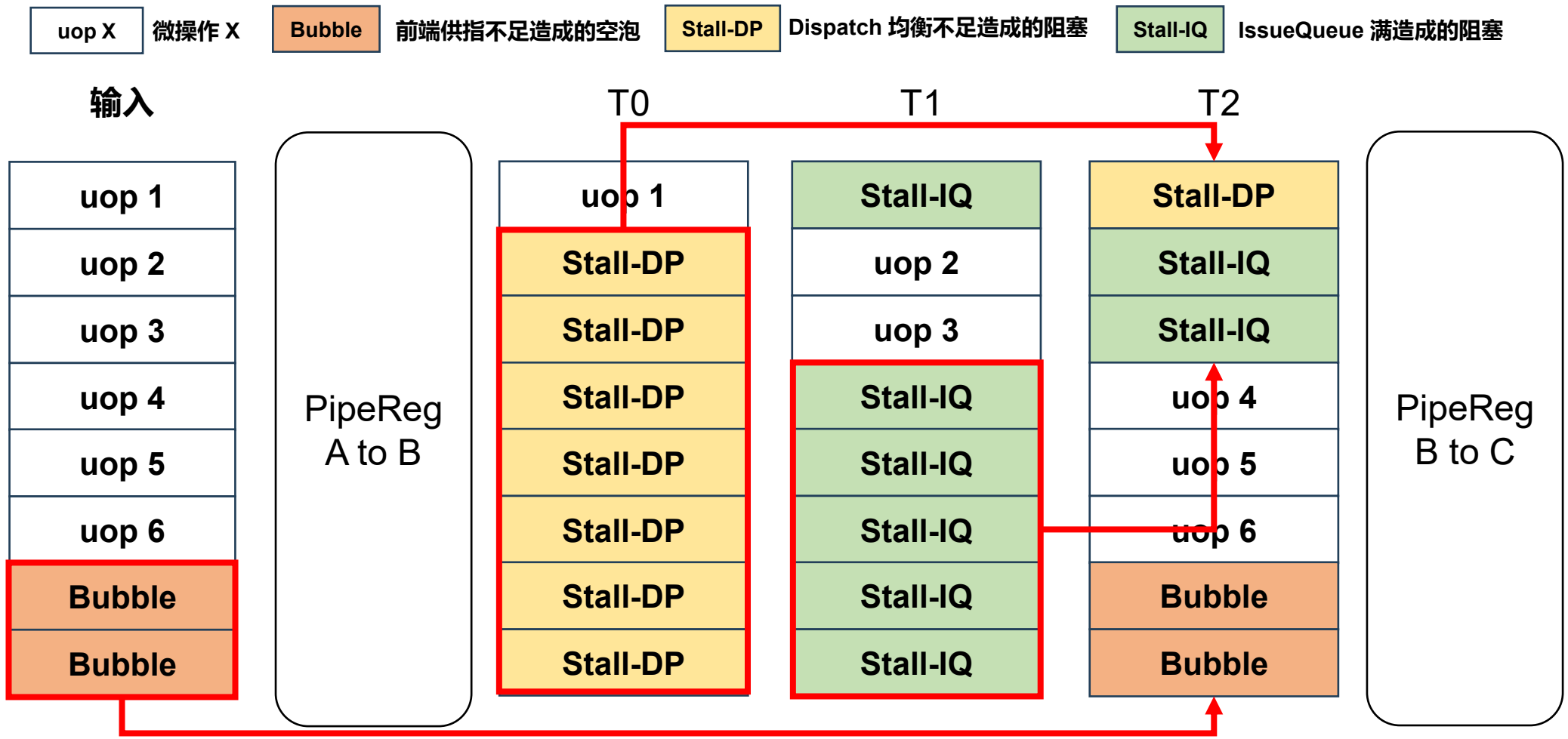
03 逐槽归因

在观察点读取槽位原因，汇总各类 Stall 和 Bubble

避免 reason 跨级错配

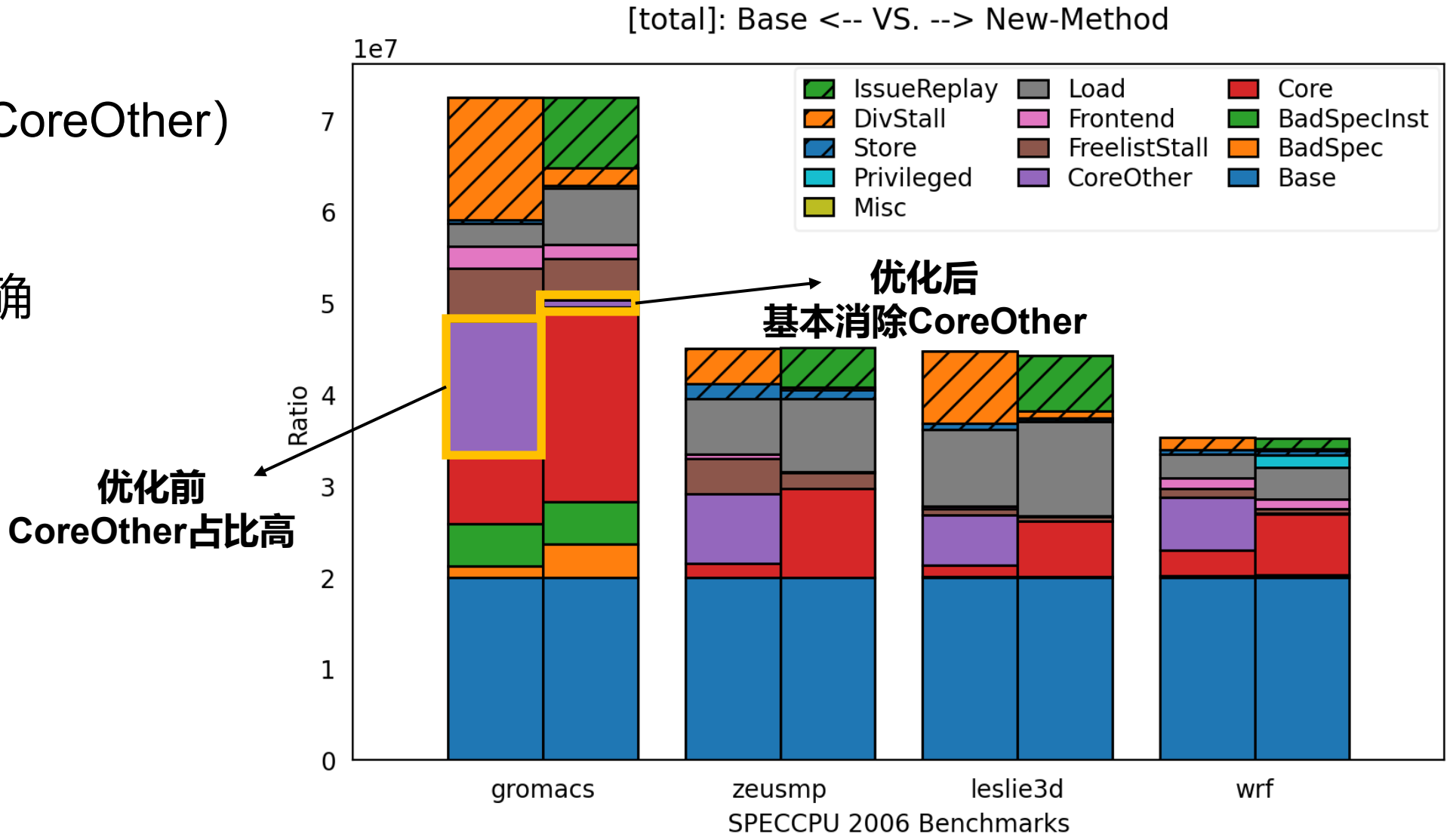
- **改进效果** 避免后级阻塞引起的归因错配，补齐更多 Stall / Bubble 的归因

示例：取指空泡与分派阻塞混合场景的归因过程



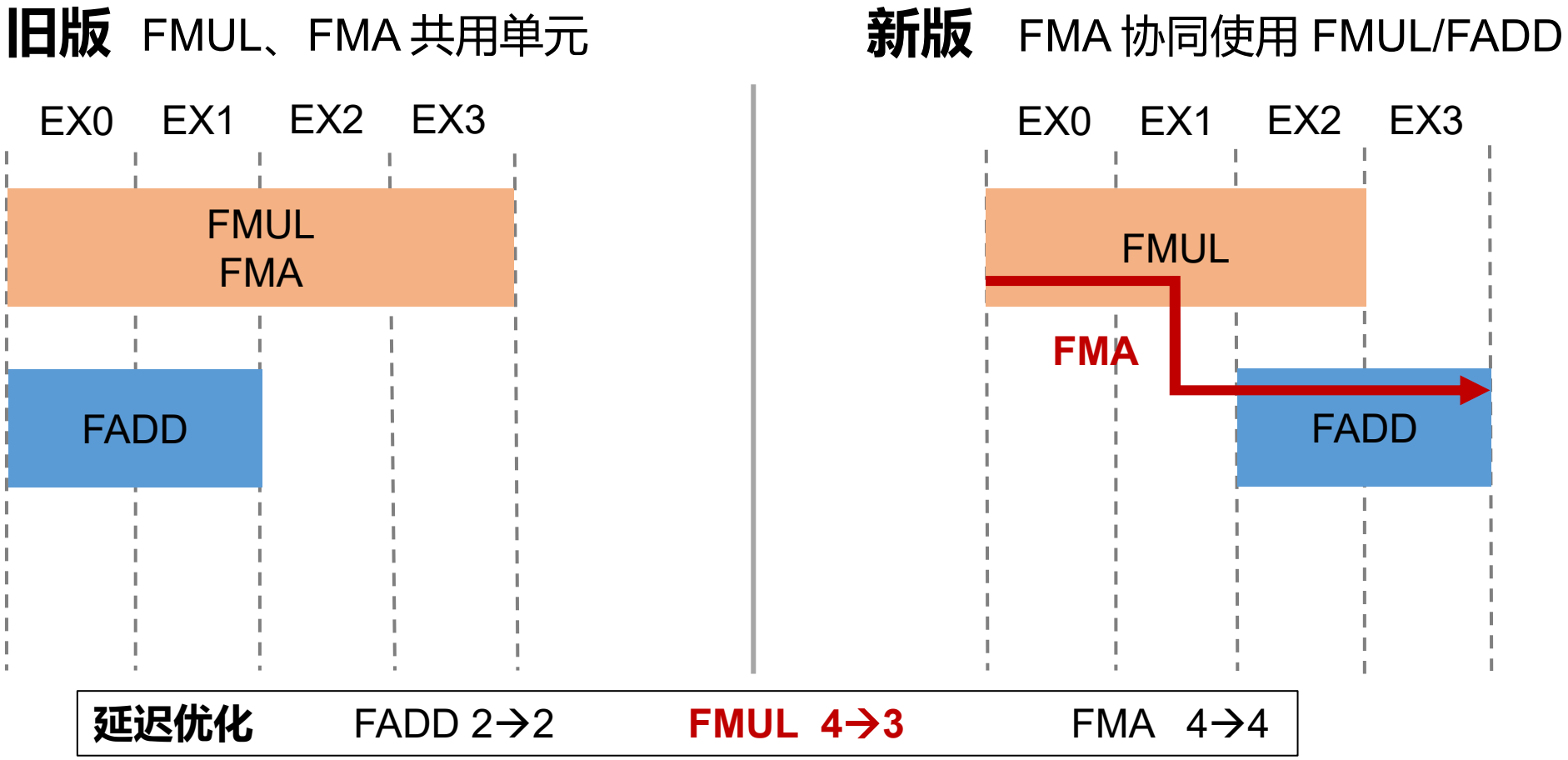
TopDown 修改前后变化

- 核心变化
  - 未知归因（CoreOther）几乎消除
  - 归因更加准确



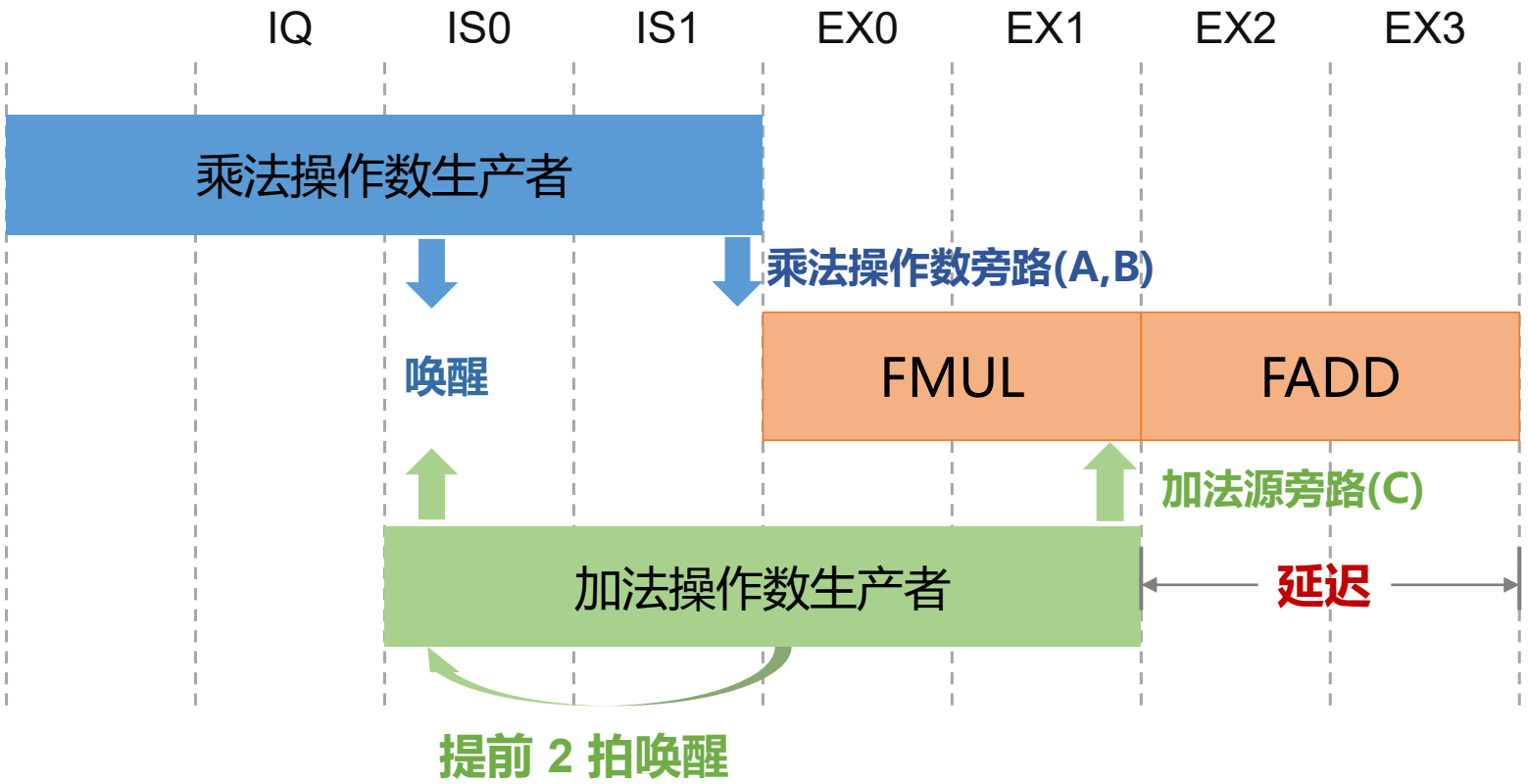
计算单元 – 浮点乘法延迟优化（FMUL）

- 瓶颈：乘法与乘加共用执行单元，拉长了乘法的延迟



计算单元 – 乘加的加法操作数延迟获取

- 观察：乘加操作先乘法后加法，无需立即获得加法操作数
- 策略：加法生产者提前 2 拍唤醒乘加指令，降低延迟至 FADD（2拍）



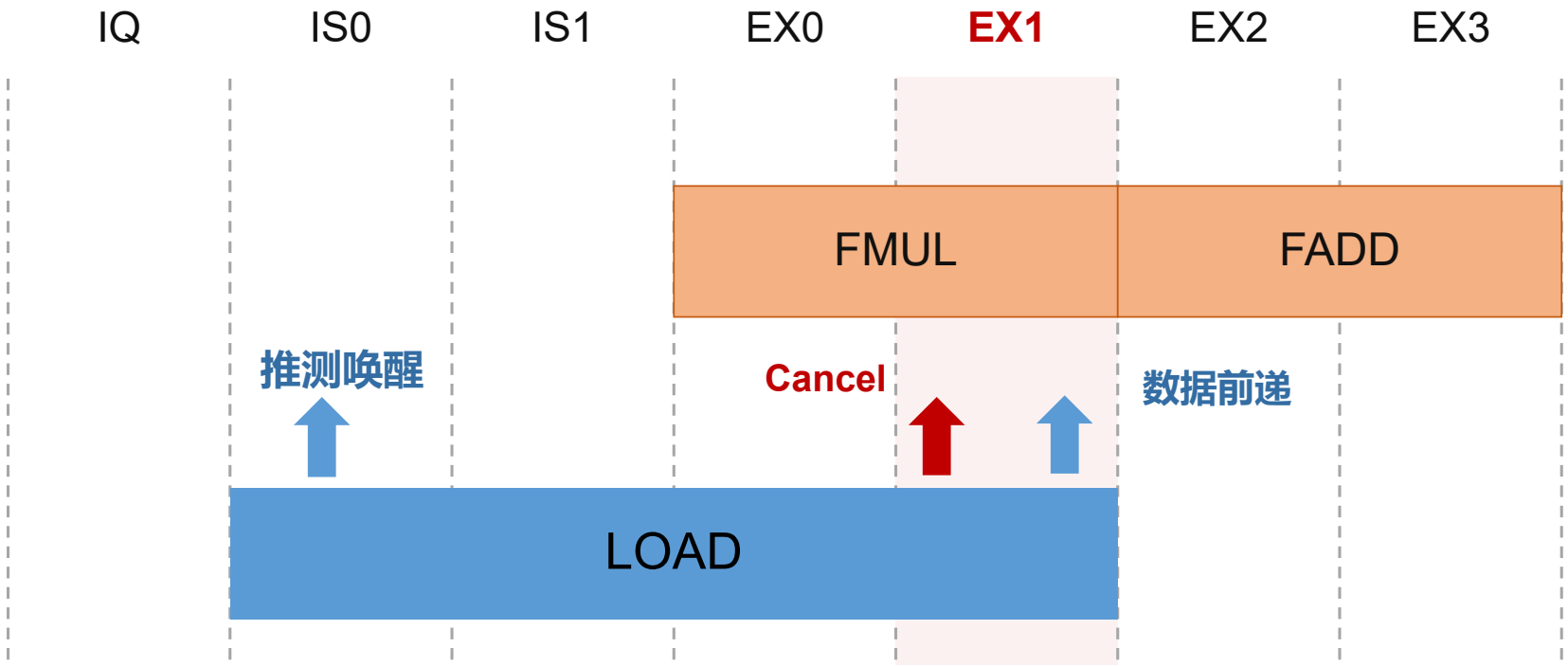
A x B + C

SPECfp2006

+3.4%

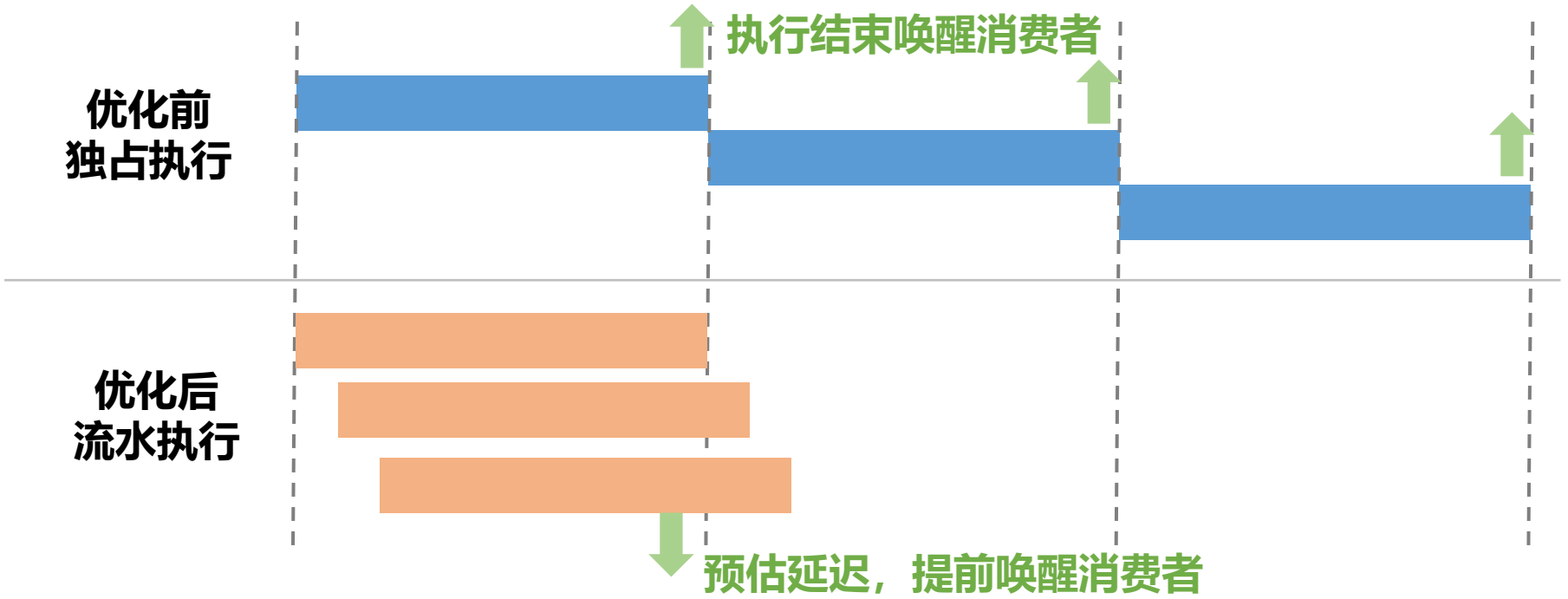
计算单元 – 激进推测唤醒的取消机制

- 难点：Load 最晚取消时机与数据执行的最后阶段相同
- 非免费午餐：推迟 FMA 的最晚被 Cancel 时机，**从 IS1 到 EX1**



计算单元 – 流水式浮点除法开方单元

- 旧版：多周期形式，吞吐为 1 / Latency
- 新版：流水线形式，吞吐为 1；提前唤醒，降低实际延迟



SPECfp2006

+3.6%



向量单元

10 月 20 日 15:45-16:00
水仙厅 · 6F | 数据中心与高性能计算分论坛
胡轩 《面向通用高性能计算的向量扩展设计》

总结

- **增加带宽**：8 路配置
- **减少空泡**：无空泡多操作译码
- **增加窗口**：混合 ROB 压缩方案
- **指令调度**：改进 Top Down 性能归因
- **执行效率**：优化浮点乘法、乘加、除法的延迟吞吐
- **向量扩展**：完全重构，优化译码、调度、计算、访存

敬请批评指正