



昆明湖架构 Coupled L2 缓存 设计与实现

陈熙¹ 张林隽² 王凯帆¹ 蔡洛姗¹ 李昕¹

¹中国科学院计算技术研究所

²北京开源芯片研究院

2023年8月24日 @第三届 RISC-V 中国峰会

背景

• 内存墙矛盾

- 内存的性能增长速度远低于处理器
- 程序员想要低延迟地访问大量的内存
- 基于时间 & 空间局部性，设计缓存



• 缓存技术的演进

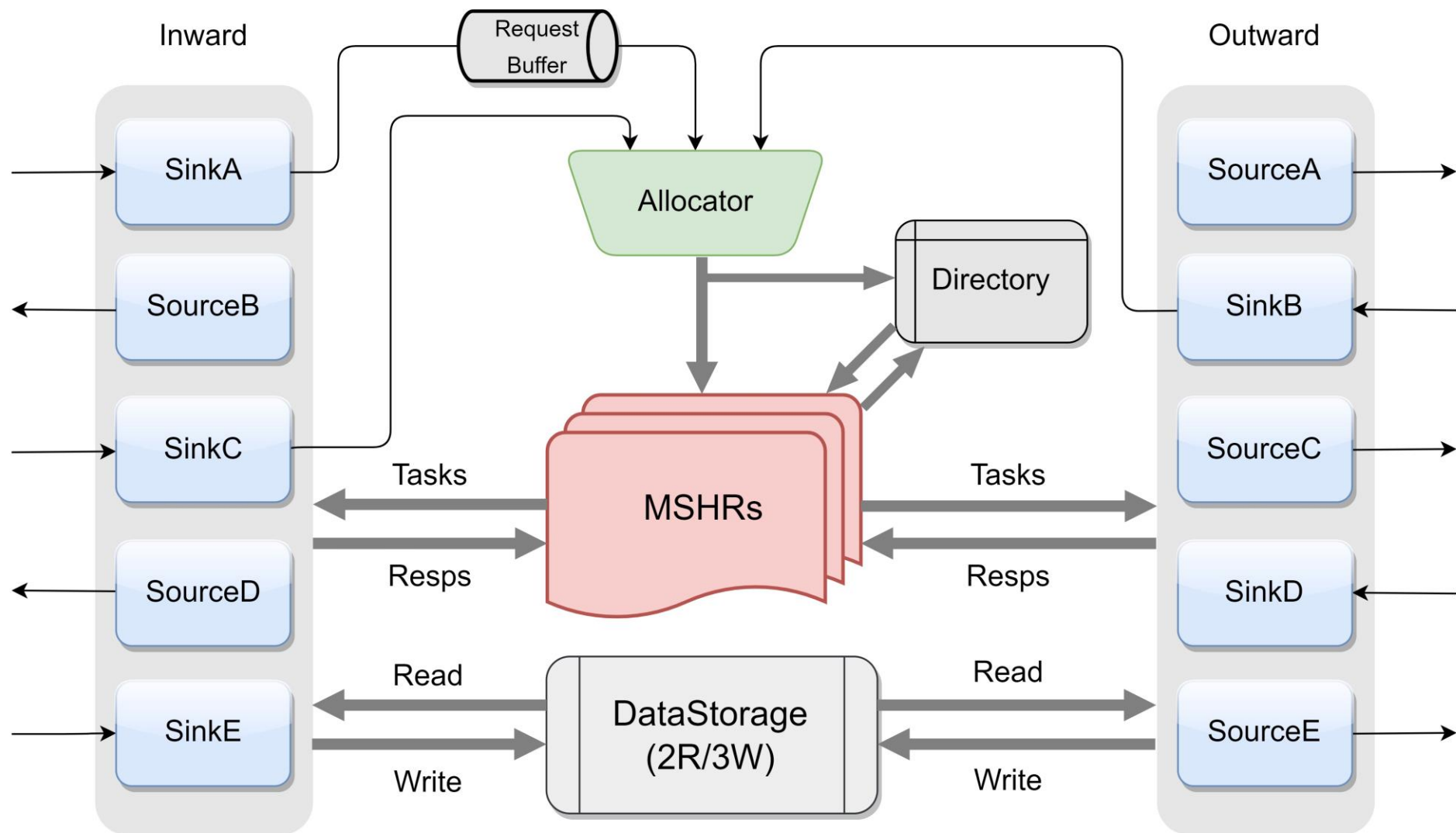
- 单级缓存 → 多级缓存
- Blocking → Non-Blocking (MSHR)
- 替换算法、预取算法



核心目标:

- 降低延迟
- 提升带宽

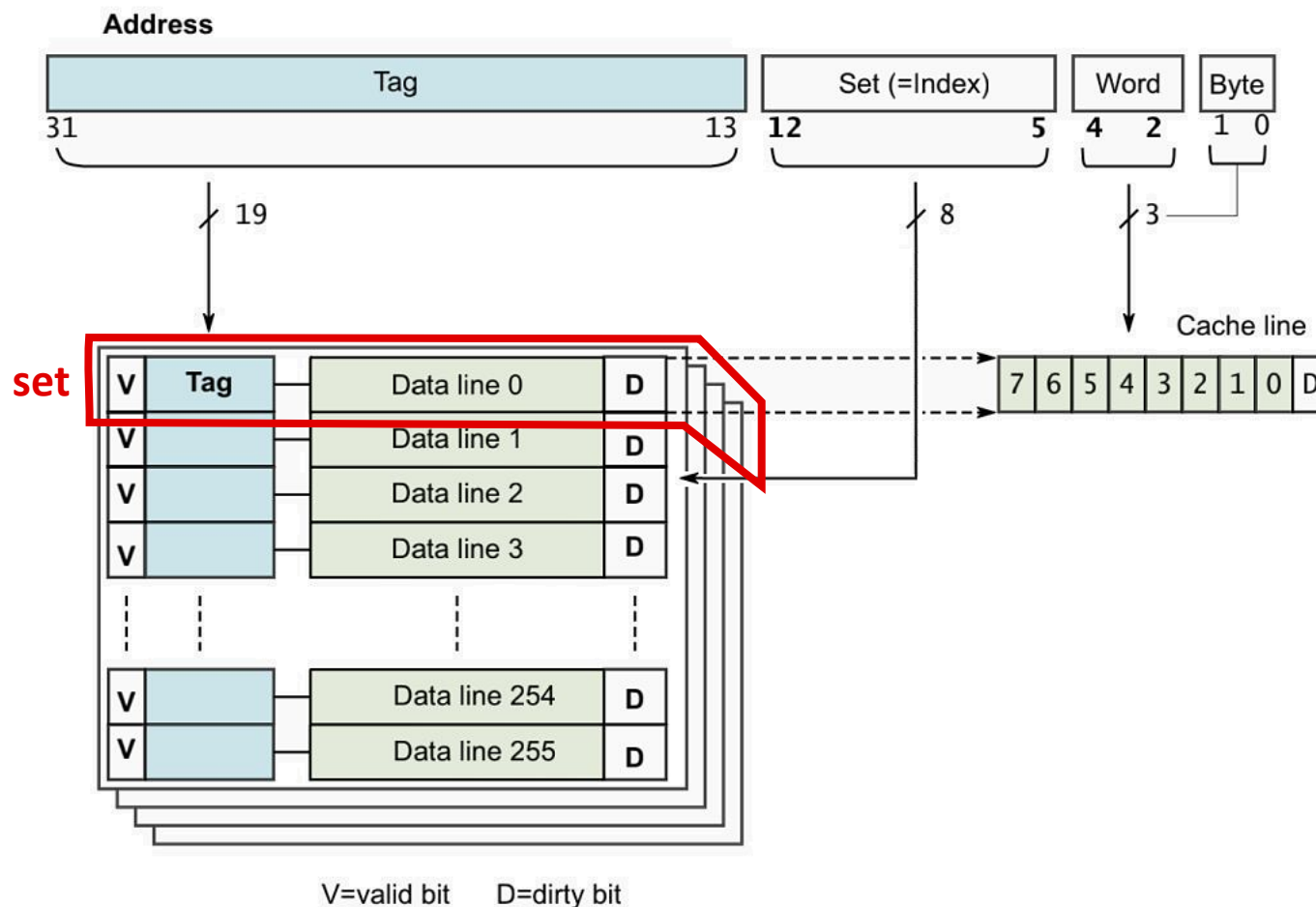
南湖 L2 HuanCun 架构回顾



南湖 L2 存在的问题 ②

• 同 Set 请求只能串行处理

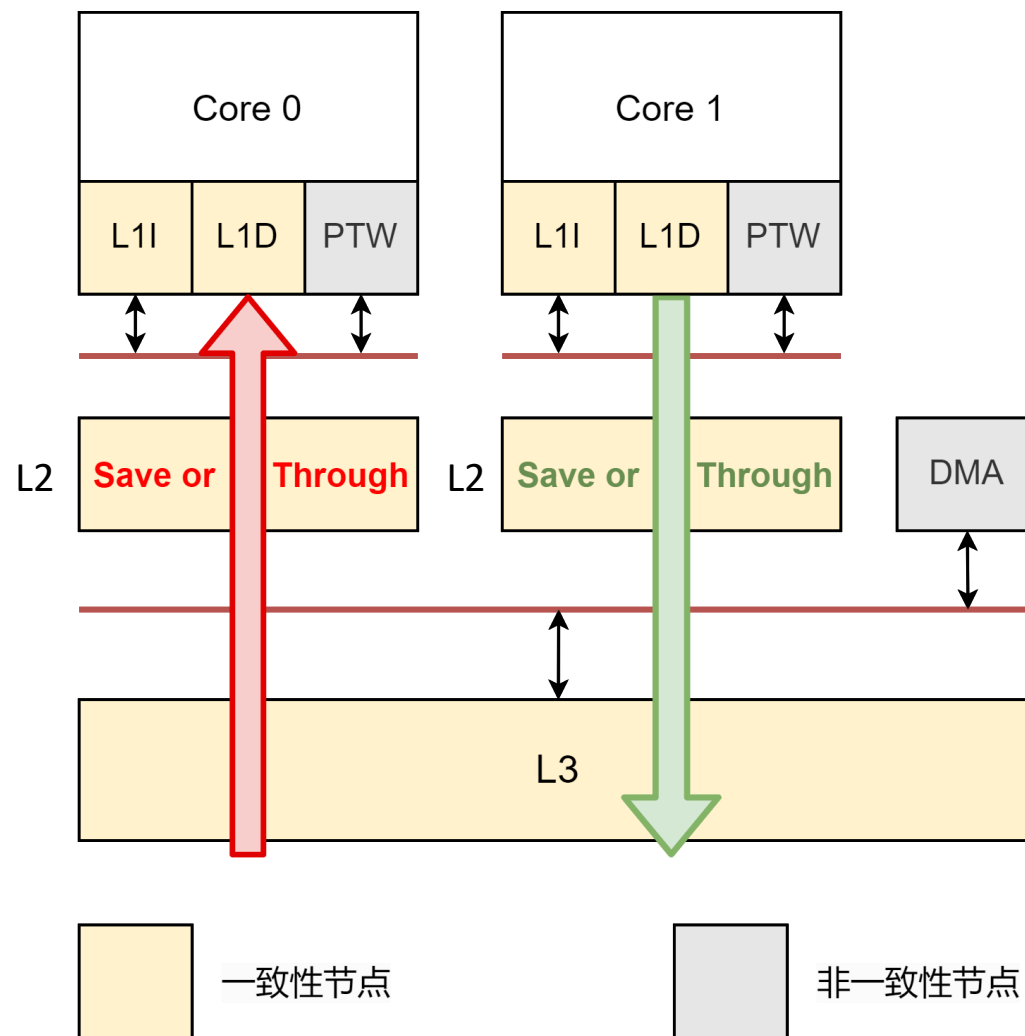
- 同 Set 会相互影响
- 为了简化设计,
限制只能同时处理一个
- 对于某些访存 pattern,
会造成严重性能损失



南湖 L2 存在的问题 ③

• 验证复杂度爆炸

- L2 采用 Non-inclusive 策略
- 仍然需要维护上层目录信息
- 有着更加多样的一致性策略



南湖 L2 存在的问题 ③

• 验证复杂度爆炸

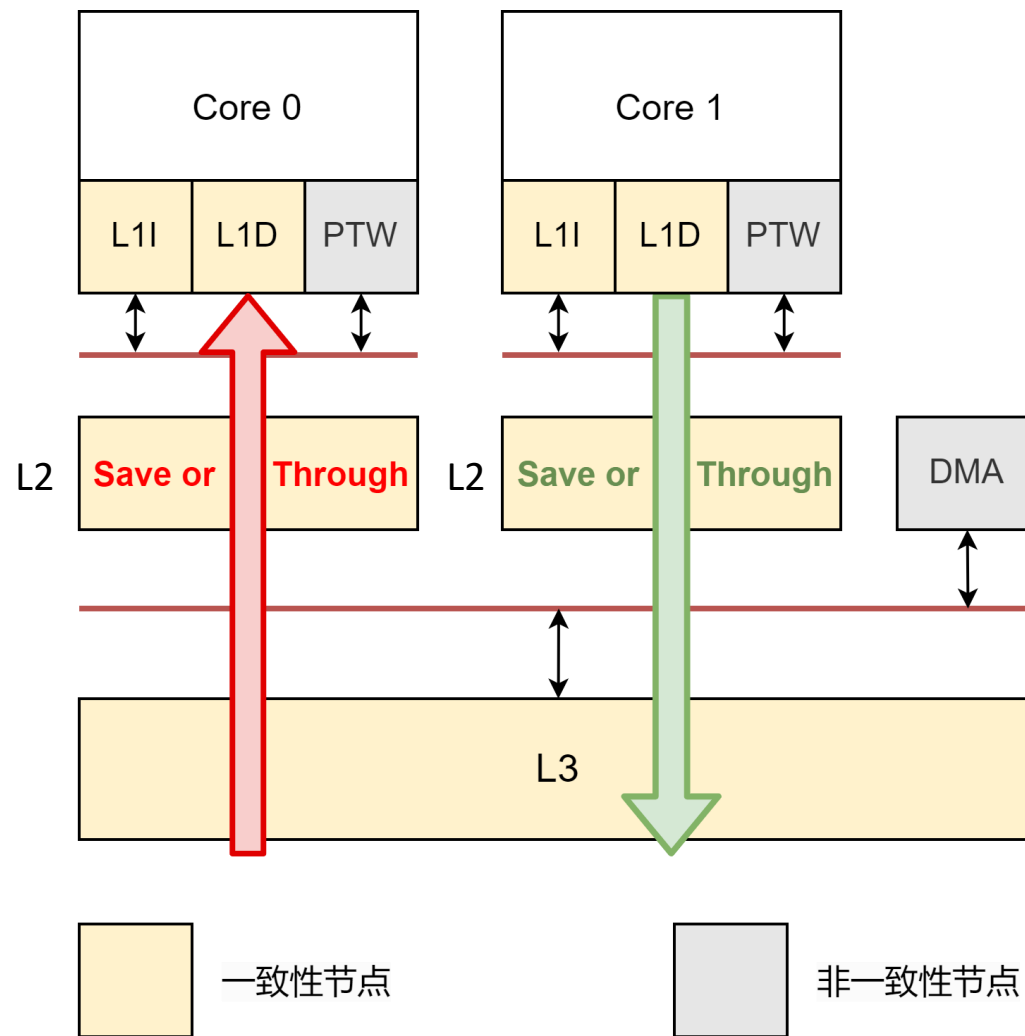
- L2 采用 Non-inclusive 策略
- 仍然需要维护上层目录信息
- 有着更加多样的一致性策略



- 需要维护 I/D 一致性



- 导致 corner case 众多，验证困难



南湖 L2 存在的问题 → 新 L2 的优化

- Hit latency 较长



采用主流流水线架构

- 同 Set 请求只能串行



同 Set 请求可以并行

- 验证复杂度爆炸

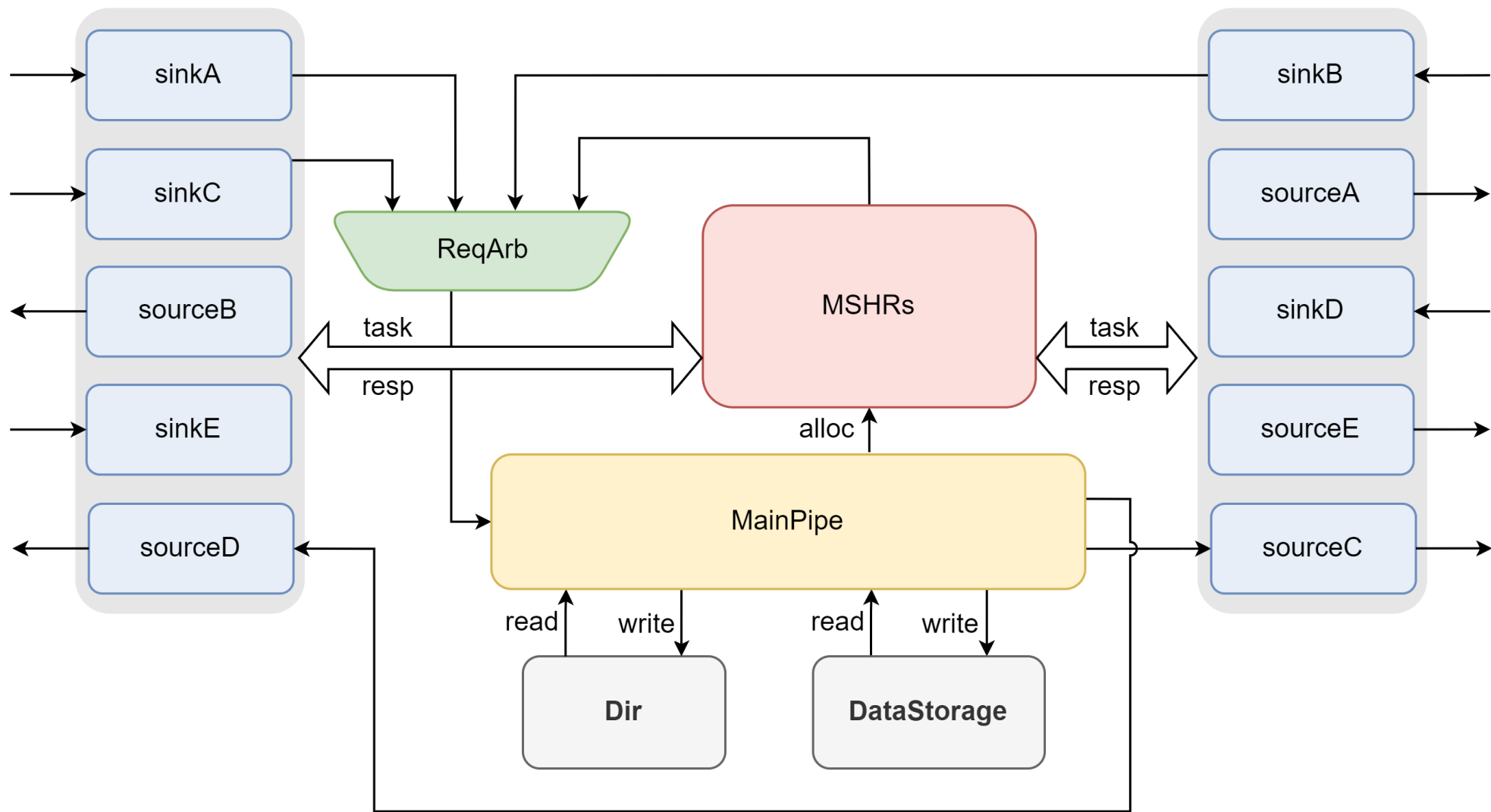


**{ Inclusive 策略
ICache 为非一致性节点**

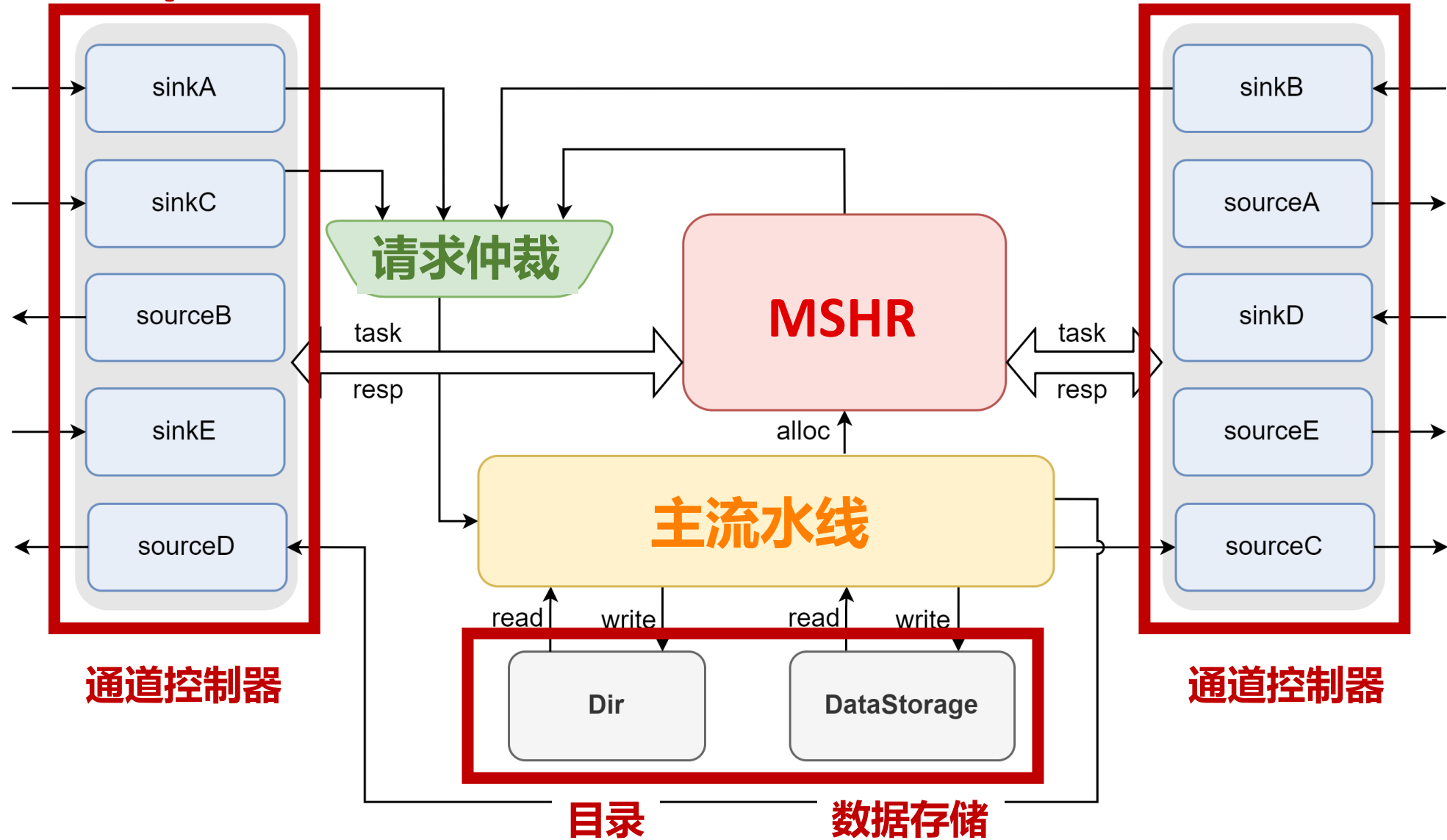


++ New Features

香山 CoupledL2 硬件结构

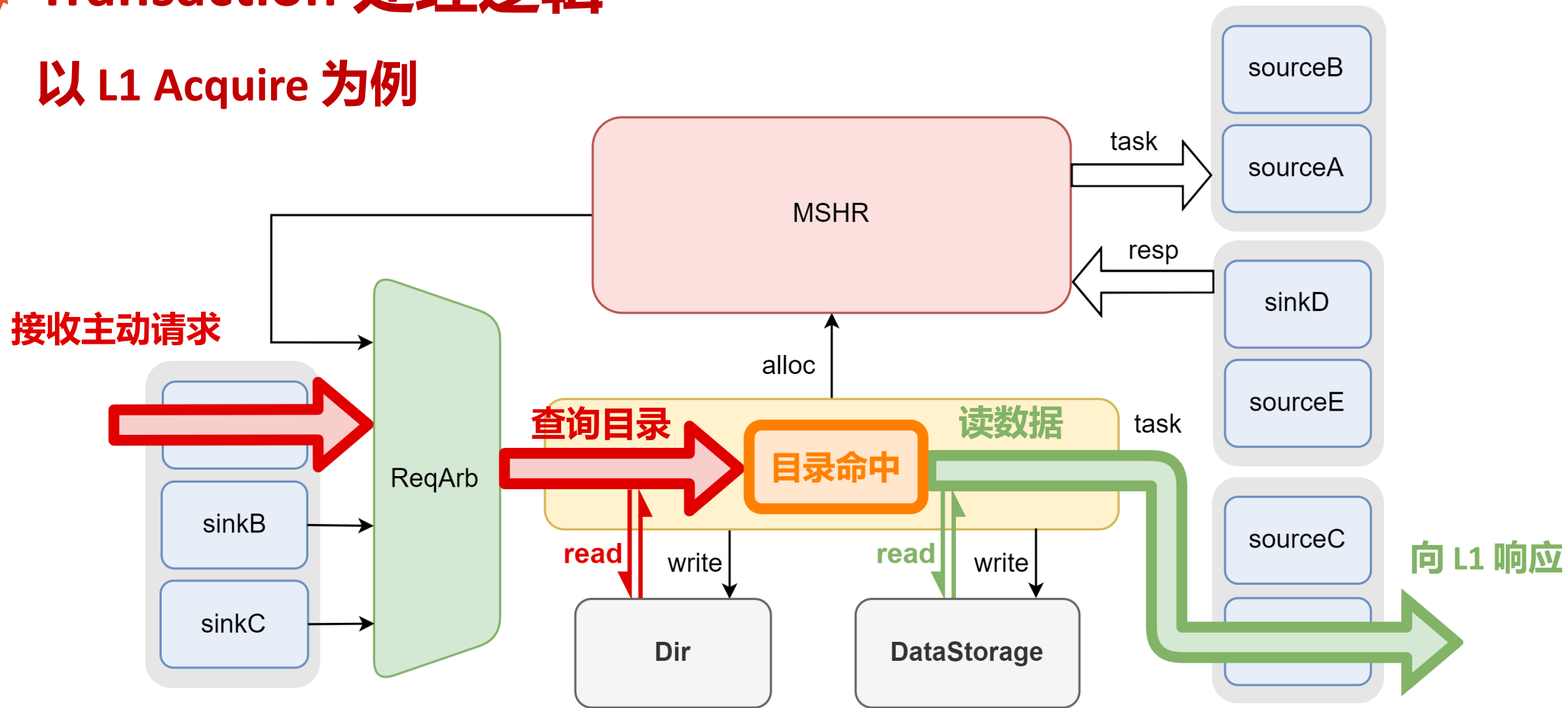


香山 CoupledL2 硬件结构



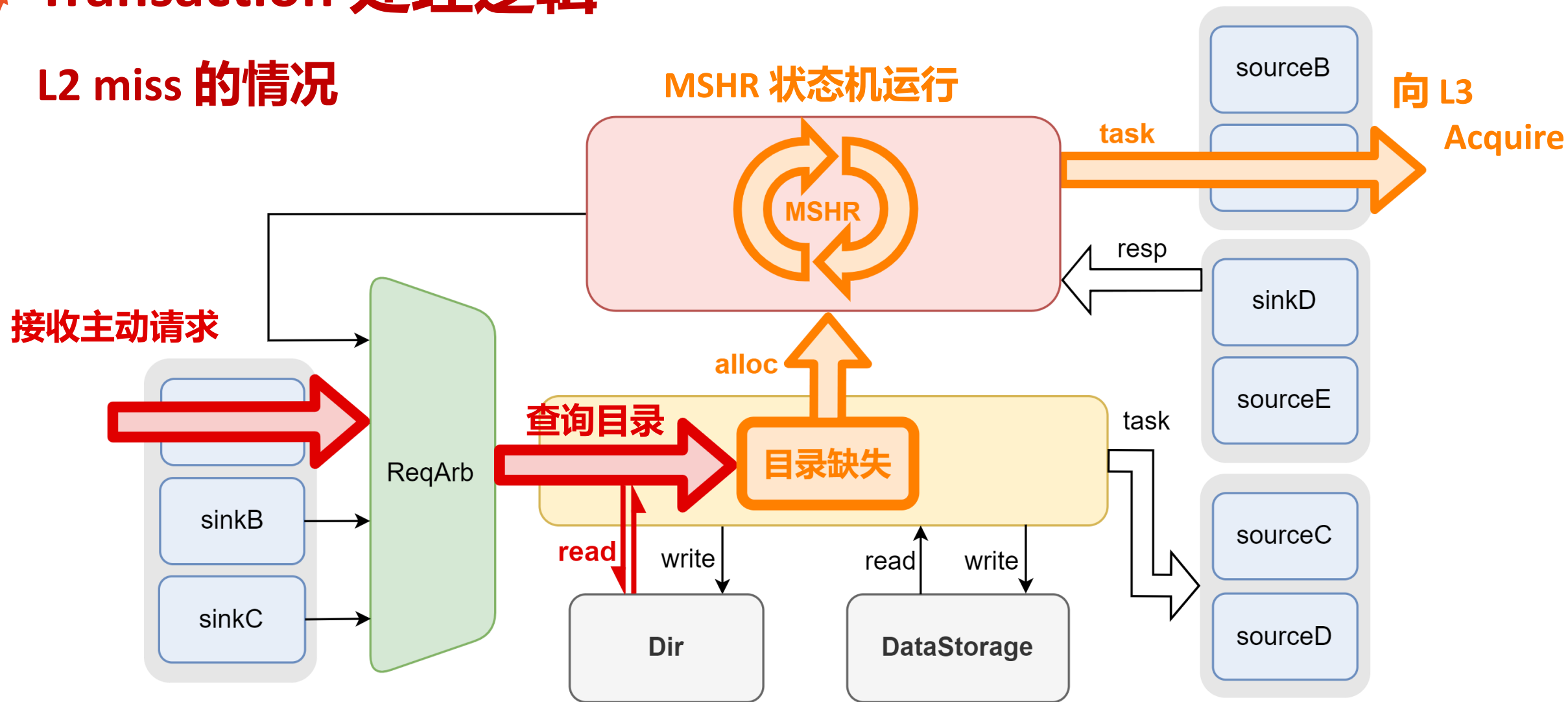
Transaction 处理逻辑

以 L1 Acquire 为例

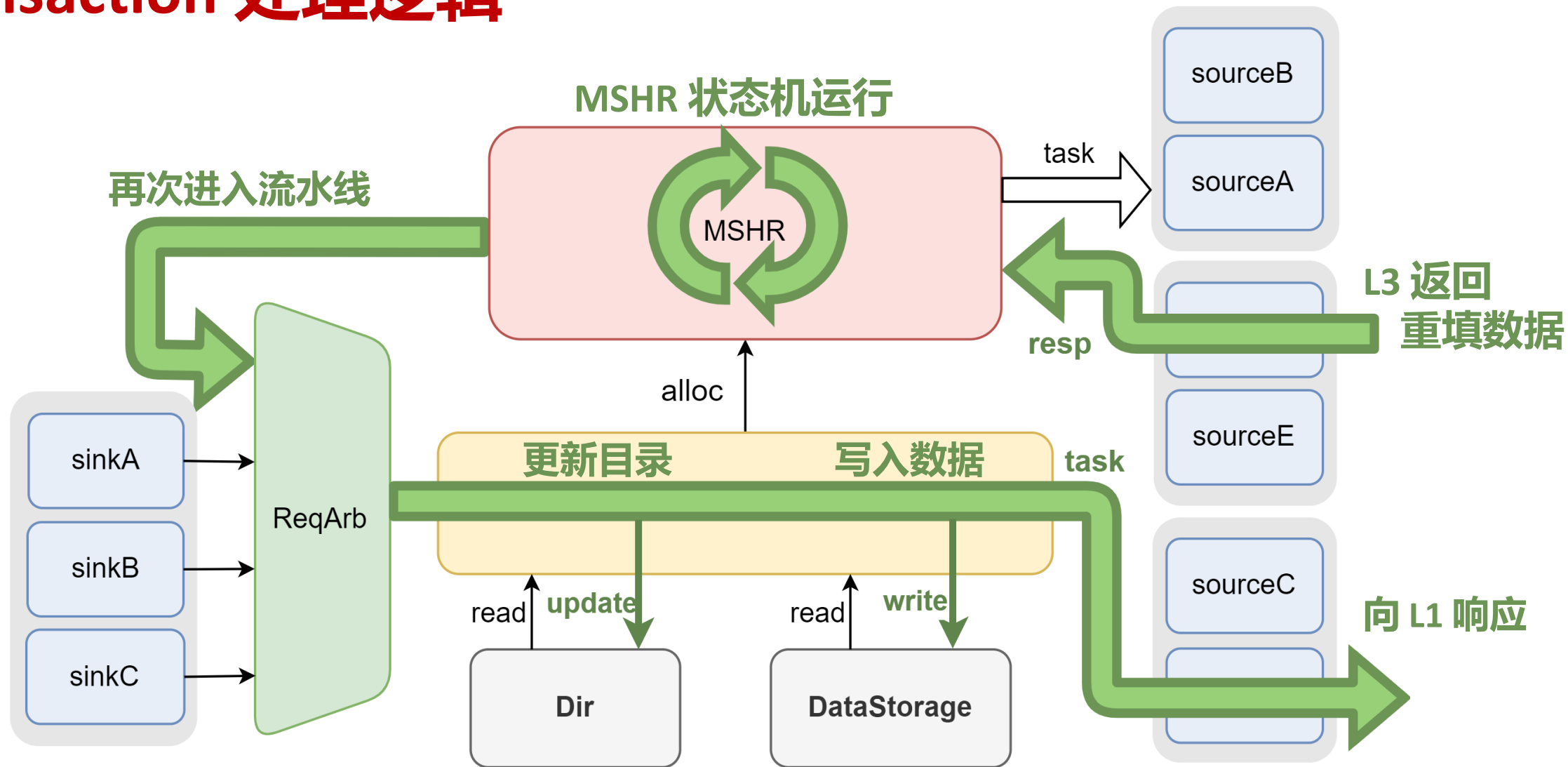


Transaction 处理逻辑

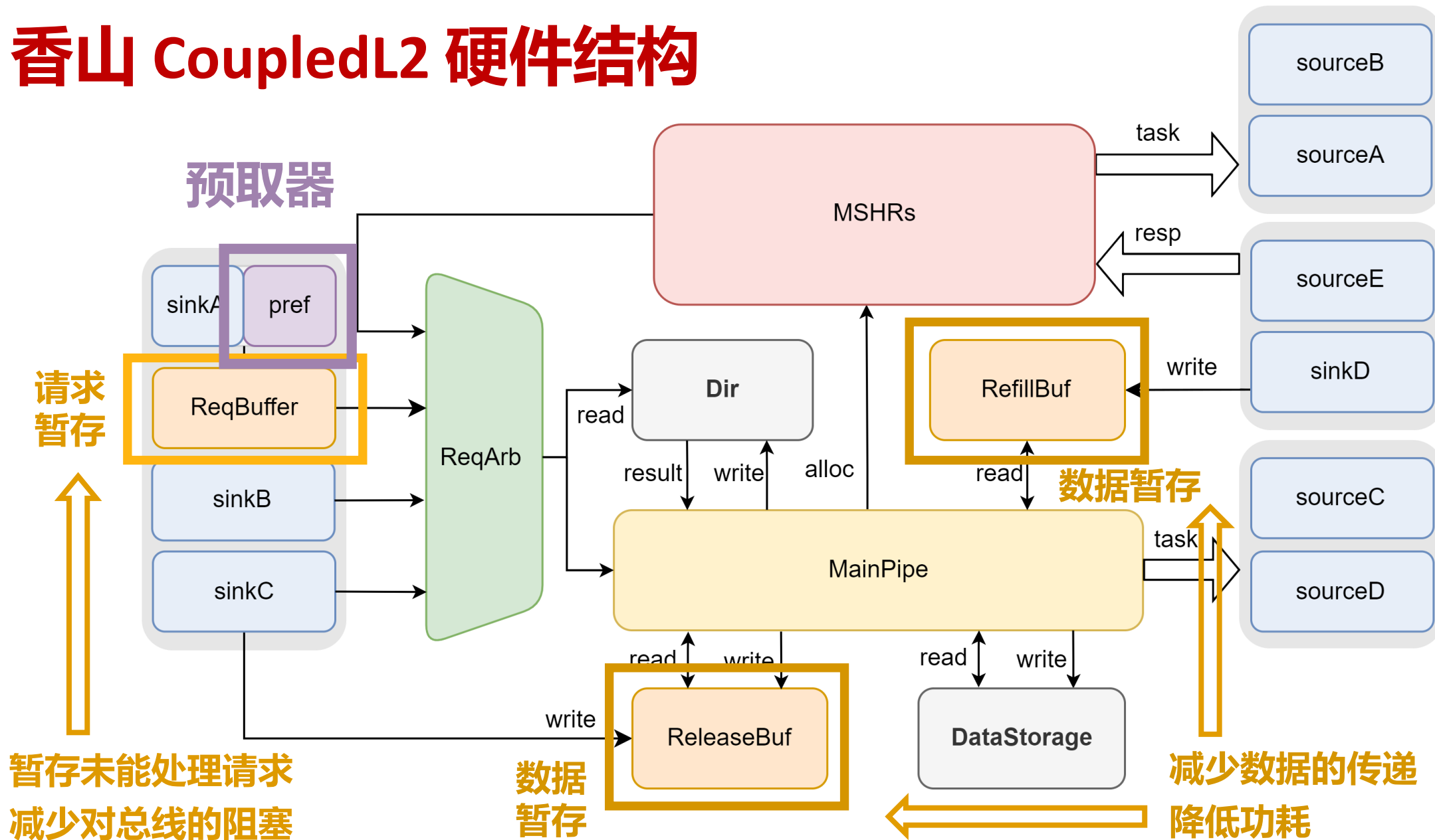
L2 miss 的情况



Transaction 处理逻辑




香山 CoupledL2 硬件结构



设计点 1 —— L1-L2 协同优化

• 观察:

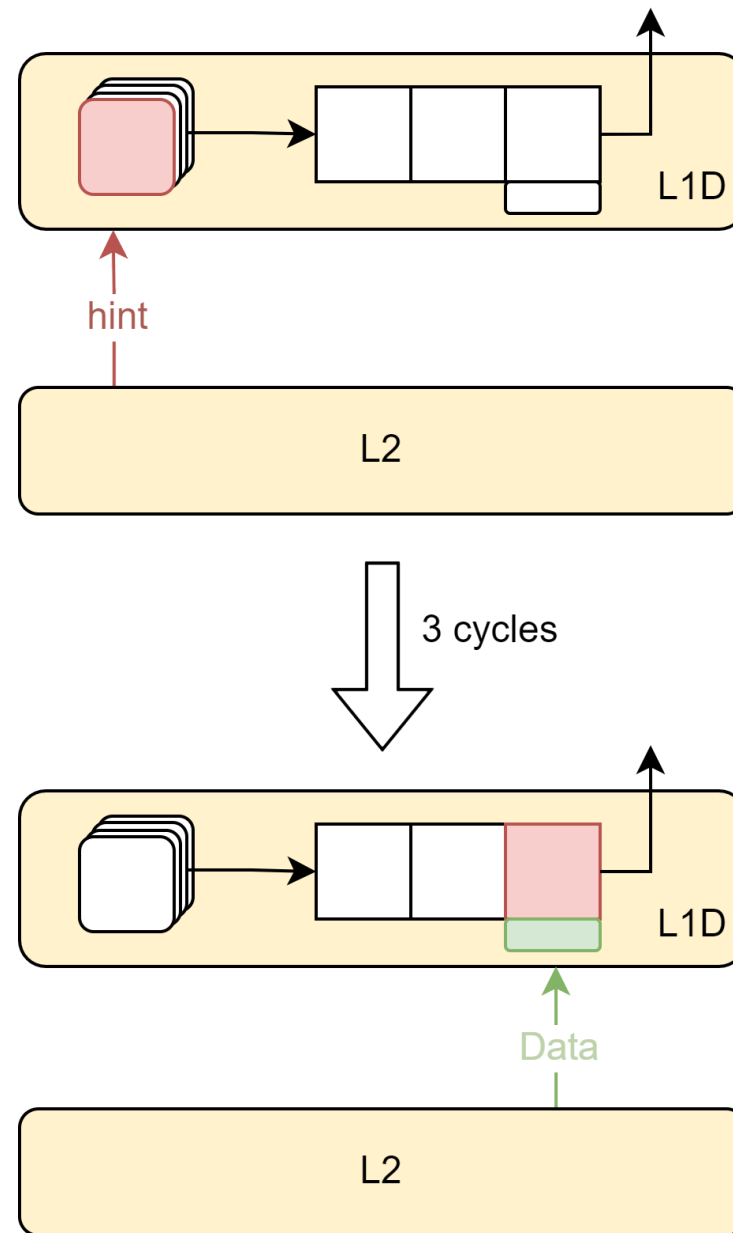
- 访存单元 miss 时存入 Load Replay Queue (图中 )
- 在 收到重填信号 和 需要用到数据 之间相隔 3 拍

• 新设计:

- 让 L2 提前给出 Hint 信号
- **加速 Load Replay Queue 的唤醒和重发**

• 具体做法:

- 设置一个 Monitor 监测主流水线
- 在 Refill 前 3 拍向 L1 发送 Hint 信号
- 需要综合考虑可能的延迟, 计算准确的 Hint 时机



设计点 2 —— 允许同 Set 请求并行处理

- **背景：**南湖 HuanCun 对于同 Set 请求，只能串行处理

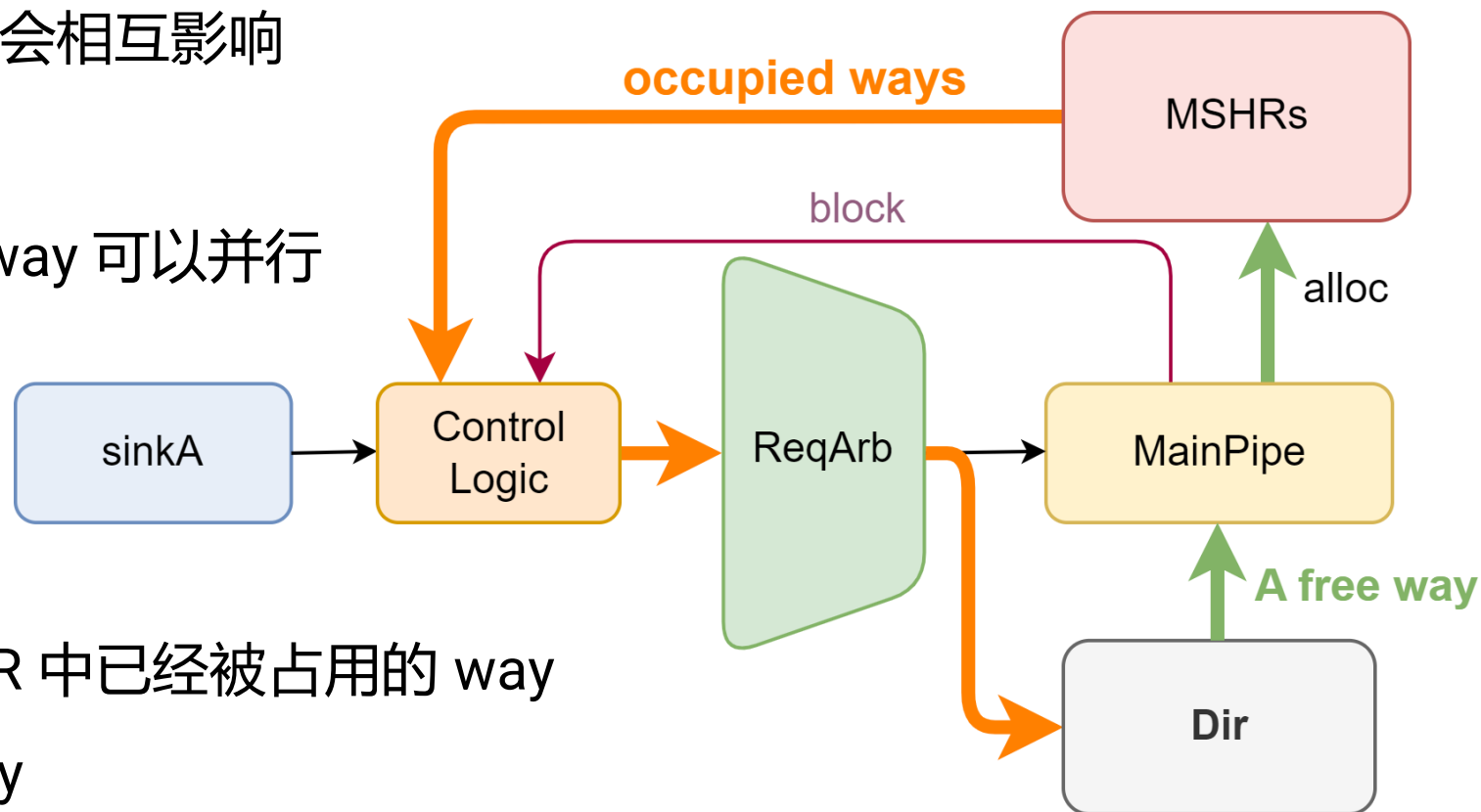
- **难点：**由于替换，同 Set 的请求会相互影响

- **新设计：**

- 同 Set 不同地址，选择不同 way 可以并行
- 只对同地址请求进行阻塞

- **具体做法：**

- 统计新请求所在 Set 在 MSHR 中已经被占用的 way
- 为新请求分配一个空闲的 way



设计点 3 —— Evict on Refill

• 观察:

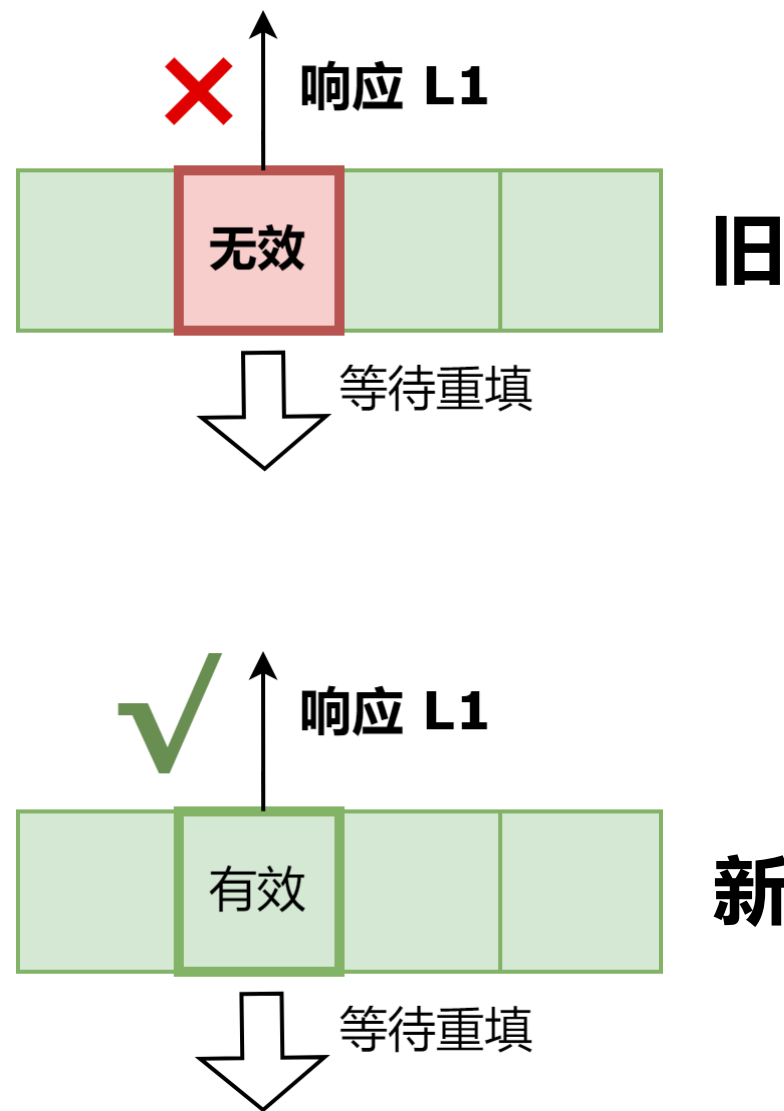
- 之前对于 L1 Acquire 缺失, 先选择一路替换旧数据
- 等待 L3 回填的时间内, 这一路被**占用**, 且无法响应 L1

• 改进:

- 在回填上来的时候才 **真正为新数据分配一路**
- 此时 L1 仍可以**命中** Set 内所有有效数据

• 具体做法:

- L1 Acquire 缺失时, 只向 L3 Acquire, 先不做替换
- 等待 L3 回填数据返回
- MSHR 向 L1 回填时, **再次读目录**并选择替换路



性能评估

- SPEC CPU 2006 测试程序
- 基于同样的香山处理器核 + L3
 - 相同的 L2 规格：1MB, 4 Slices
 - Non Incl. HuanCun L2 vs. CoupledL2
- **性能提升明显：**
 - 部分程序上有近 50% 的提升
 - SPEC 整体提升 1 分/GHz
- **时序提升：**
 - 主频 2GHz → 3GHz

SPEC分值/GHz	南湖 HuanCun	昆明湖 CPL2	提升百分比
astar	6.54	6.88	5.14%
bwaves	18.81	19.99	6.22%
bzip2	7.50	7.85	4.56%
cactusADM	12.76	13.53	6.00%
calculix	5.19	5.19	0.02%
deall	17.74	17.97	1.27%
gamess	13.38	13.36	-0.09%
gcc	12.93	15.70	21.43%
GemsFDTD	14.56	21.32	46.45%
gobmk	9.04	9.09	0.59%
gromacs	9.52	9.90	3.92%
h264ref	16.70	16.85	0.92%
hmmer	14.70	14.73	0.25%
lbm	28.01	29.81	6.42%
leslie3d	12.14	14.46	19.15%
libquantum	20.58	24.65	19.78%
mcf	10.66	9.93	-6.83%
milc	10.97	11.58	5.53%
namd	14.00	14.01	0.04%
omnetpp	7.84	8.19	4.50%
perlbench	11.45	11.76	2.68%
povray	15.77	15.81	0.26%
sjeng	9.15	9.33	1.99%
soplex	13.87	14.11	1.69%
sphinx3	12.25	18.72	52.75%
tonto	9.91	9.91	0.03%
wrf	9.19	9.27	0.89%
xalancbmk	9.96	11.74	17.86%
zeusmp	15.45	17.82	15.34%

下一步的优化设计

• 请求融合

- 对于**同地址请求**：将新收到的请求融合进 MSHR 里尚未完成的请求
- L3 无效化 L2 → L2 主动释放 => L2 直接响应 L3
- L1 Acquire → 预取 => 预取同时回填 L1

• 关键字优先

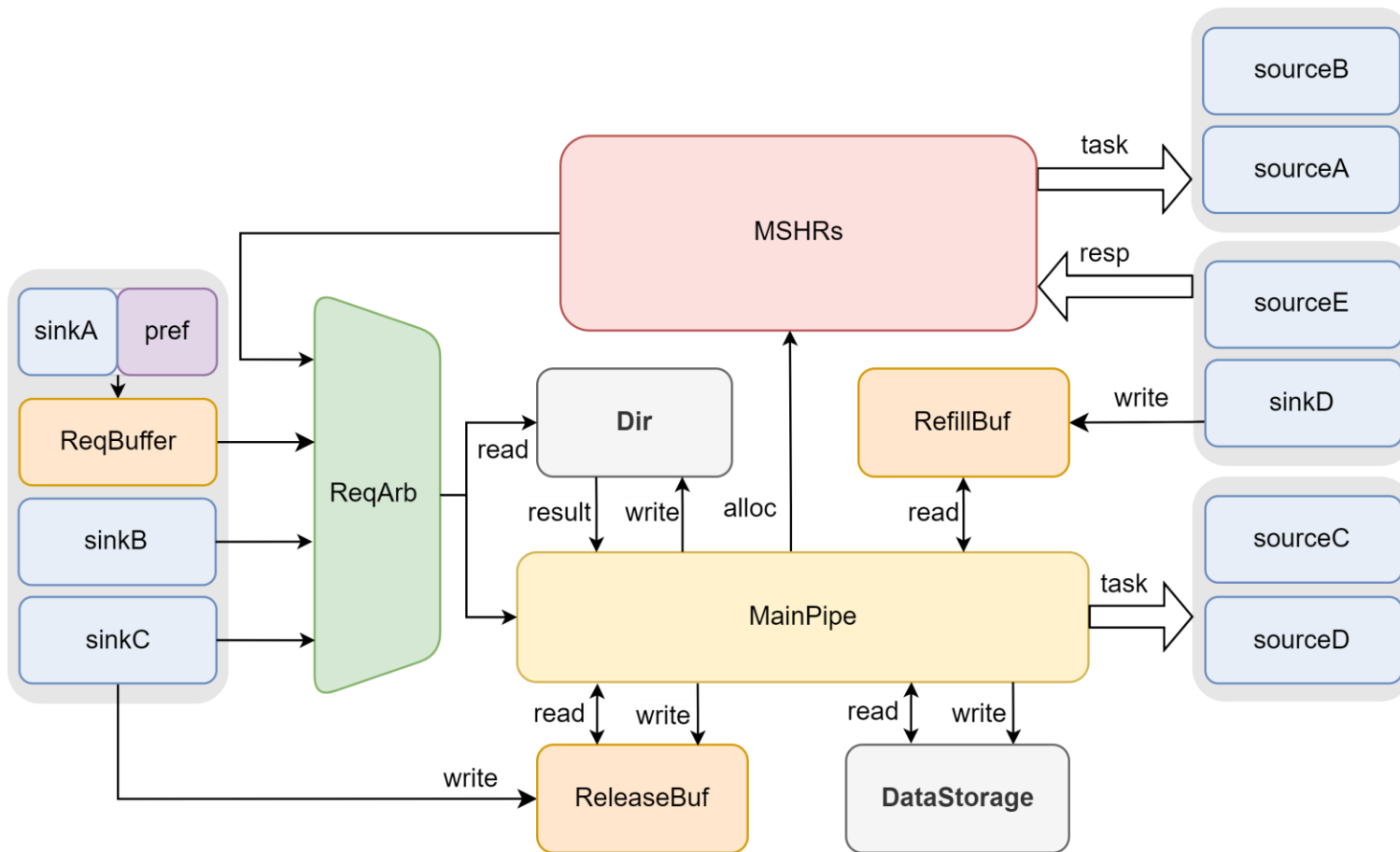
- 目前一个缓存块需要两次总线传输 (2 beats)
- 向 L1 重填时优先发送【触发 miss 地址】所在的 beat

• 重填数据 bypass

- 从 L3 重填回来的数据不经过主流水线，直接发送给 L1

🌟 总结：香山新 CoupledL2 的设计与实现

- 南湖 L2 存在的问题
- CPL2 整体设计
 - 非阻塞式主流水线
 - L1-L2 协同优化
 - 同 Set 请求并行处理
 - Evict on Refill
- 性能评估
- 下一步优化设计





谢谢！ 敬请批评指正！

- 南湖 L2 存在的问题
- CPL2 整体设计
 - 非阻塞式主流水线
 - L1-L2 协同优化
 - 同 Set 请求并行处理
 - Evict on Refill
- 性能评估
- 下一步优化设计

