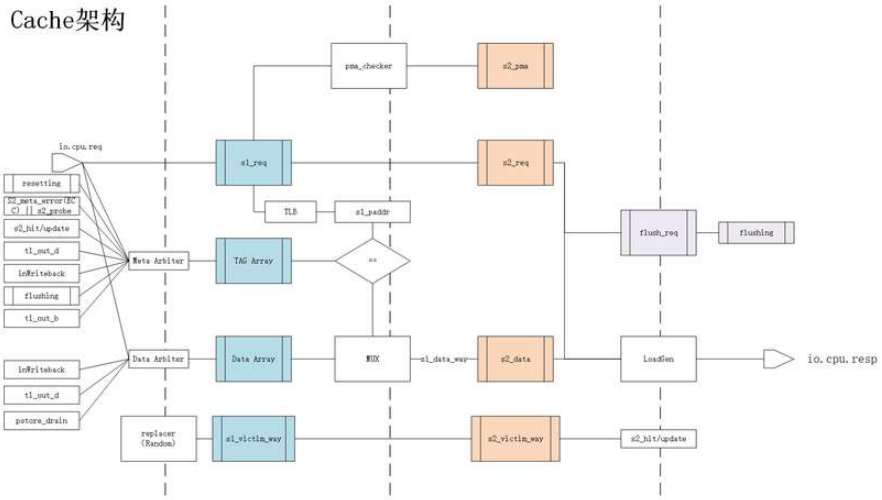
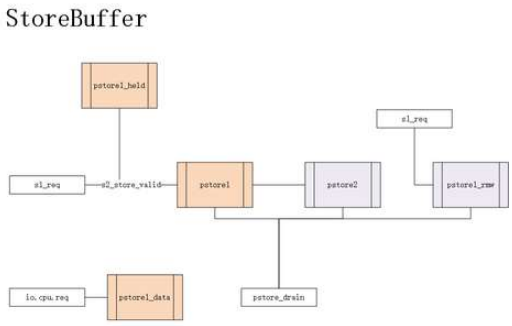
Rocket-DCache

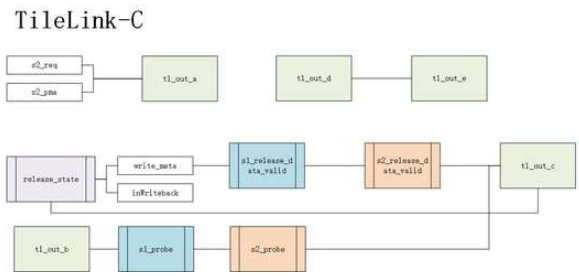
总体架构

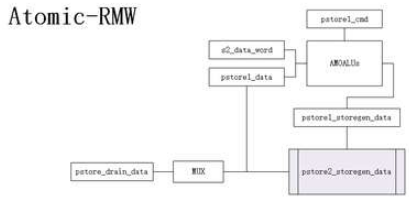
RocketChip DCache流水化访问分为三个阶段，有两个单元的StoreBuffer（pstore1与pstore2，2优先级大于1）

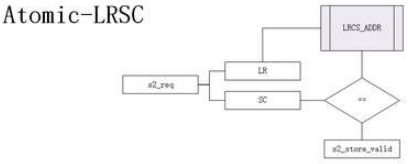
- T0：CPU发出请求，DCache访问Meta(Tag+Coh)与Data
- T1：访问L0 TLB与PMA，并对比Tag与paddr
- T2：输出访问结果











Meta读写端口

1. resetting **Write**

复位后resetting置位，启动计数器flushCounter，将所有Meta写为0（Nothing）

2. s2_meta_error **Write**

meta ecc纠错，如果遇到ecc无法纠正的错误，则将此meta复位

3. s2_update **Write**

访问命中，并且meta一致性状态需要转变（写请求导致状态从trunk到dirty），可被cpu.s2_kill取消

4. refill **Write**

当tl_d返回数据时，更新meta

5. release **Write**

释放块时需要更新meta权限，由release状态机控制

6. flush **Read**

冲刷所有meta时，通过flushcounter计数器遍历所有meta，将meta读出，通过release释放权限

7. Tilelink b通道 **Read**

响应tilelink b通道发来的Probe请求，probe消息会进入s1阶段，此时先读取当前meta供s2阶段处理probe，可能需要release

8. cpu.req **Read**

响应CPU发来的读写请求

Data读写端口

1. pstore_drain **Write**

处理来自Store buffer（pstoreX）的写请求，此请求可以被cpu.s2_kill信号取消

2. refill **Write**

当tl_d返回数据时，更新data

3. inWriteback **Read**

在release处理过程中，读取Data，将脏数据写回

4. cpu.req **Read**

响应CPU发来的可能需要读取的请求

访问过程

- T0：判断是否可以接受请求，读Meta与Data

忙情况：

meta或data不可响应CPU请求

需要查询tlb并且tlb未ready

release处理

有在等待grant的缓存请求

需要查询tlb/ptw且miss/busy

外部与CPU同时访问tlb导致tlb冲突，优先响应外部请求

未能处理请求(nack)

load与store buffer冲突（RAW）

处理probe请求

- T1：查询TLB，与PMA，对比TAG，检查一致性权限，判断hit

TLB当周期给出结果，如果tlb miss s1_nack置位，最终nack在s2发送回CPU。如果为uncached请求，等待总线返回数据后，向CPU发起replay_next，CPU阻塞一拍，将数据写回寄存器并修改记分板。

- T2：将访问结果返回CPU，命中则需要对数据通过loadGen进行signed/ext处理。

IO MSHR

仅uncached请求会使用MSHR，s2发出uncache'd请求时，分配一个MSHR保存该请求，当tl_d返回数据时，清除已经分配的MSHR。uncached请求不会返回nack。

通过tilelink d返回的的source区分不同MSHR，如果是uncached授权，则清除对应的MSHR

分配的MSHR用于检查IO请求可能发生的WAW RAW WAR冲突

如果MSHR不为空，Cache无法发出cached miss请求（tl_a），无法开始Flush处理

如果MSHR无空闲，无法处理uncached请求

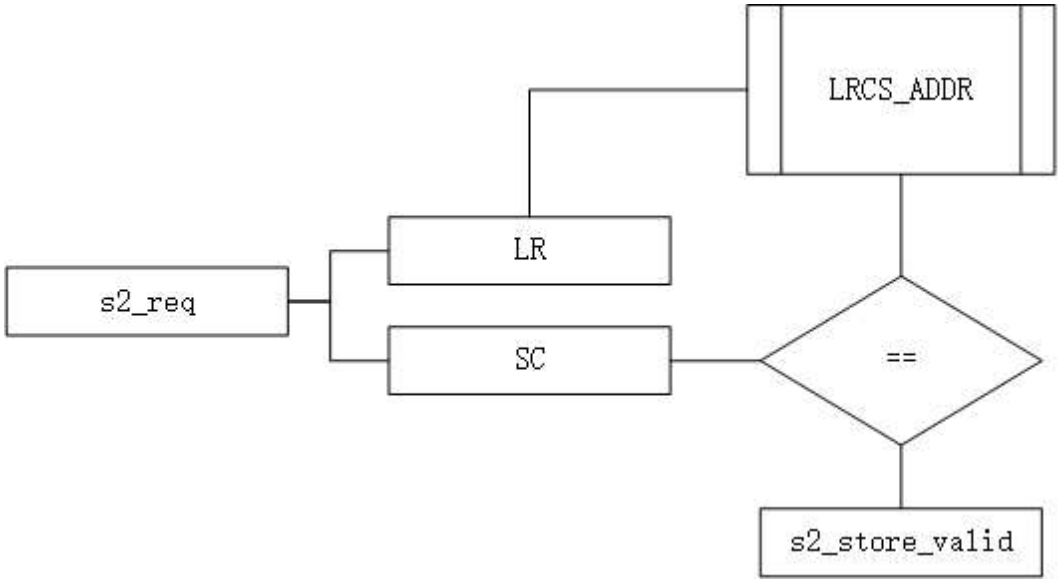
OutStanding

io.cpu.ordered信号报告CPU是否有在执行的访存请求，为0表示有，1表示无

Atomics

LR/SC

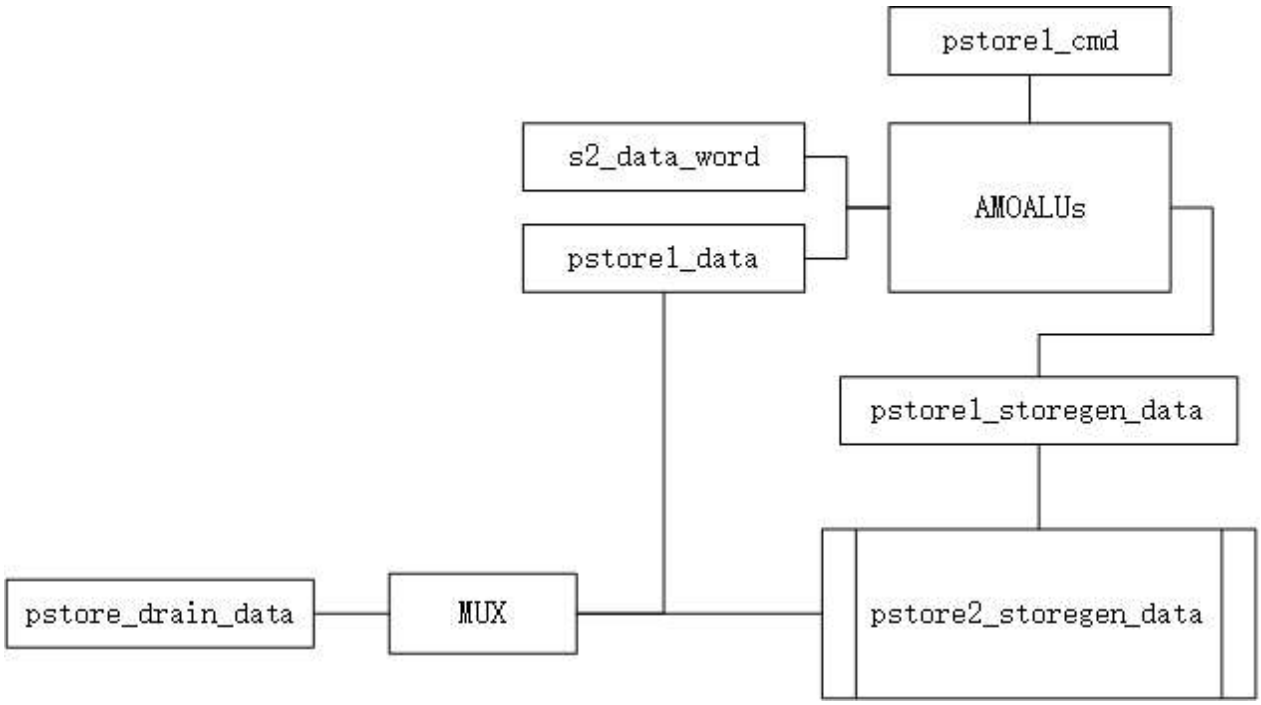
使用LRSC_ADDR寄存器存储LR地址，在进入Store Buffer前检查地址



RMW

在收到RMW类型原子指令时，s1阶段读出Meta与Data并判断hit，如果未能hit则按照普通访问miss情况处理（nack replay），s2阶段置位pstore1_rmw寄存器表示该操作为RMW类型原子访问

注意：原子操作RMW只有在store buffer的最后一项（pstore2）才可以写回



CPU端侧处理

如果检测到Outstanding（io.cpu.ordered信号）有效或者正在对DCache发起读写请求，则标记id_mem_busy，此信号用于fence

当Cache Miss时，此时CPU访存指令刚好进入wb阶段，wb阶段检测到DCache端口valid无效即可判断miss，此时不写回本次指令执行结果。如果本次访存为uncached则置位计分板，等待DCache返回uncached数据时，DCache通过replay_next报告CPU，否则冲刷流水线，重新取访存指令执行。