|  |  |
| --- | --- |
| 文档编号 |  |
| 文档版本 |  |
| 文档管控 | 内部公开 |
| 存档日期 |  |

昆明湖项目RS模块AS

|  |  |
| --- | --- |
| 编 写： | 胡轩 |
| 校 对： |  |
| 审 核： |  |
| 批 准： |  |

昆明湖V1项目

2024年XX月XX日

文档修订记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **版本编号** | **变化状态** | **变更说明** | **作者** | **日期** |
| 1 | V0.1 | C | 语雀文档导出 | 胡轩 | 2023.6.13 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

\*变化状态：C—创建，A—增加，M—修改，D—删除

文档审批信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本** | **审核** | **会签** | **批准** | **备注** |
| V1.0 |  |  |  |  |
|  |  |  |  |  |

目 录

[1 简介 1](#_Toc10817)

[1.1 文档介绍 1](#_Toc28428)

[1.2 参考文档 1](#_Toc26482)

[1.3 术语说明 1](#_Toc6174)

[1.4 技术背景 1](#_Toc11759)

[2 设计规格 2](#_Toc21679)

[3 功能描述 2](#_Toc18628)

[3.1 功能概述 2](#_Toc4276)

[3.2 功能详述 2](#_Toc29895)

[3.2.1 接收并暂存来自 Dispatch 的指令信息 2](#_Toc23450)

[3.2.2 维护指令源物理寄存器状态 3](#_Toc23026)

[3.2.3 选择指令发射执行 3](#_Toc17061)

[3.2.4 访存指令重发 4](#_Toc10438)

[4 总体设计 5](#_Toc29537)

[4.1 整体框图 5](#_Toc17290)

[4.2 接口列表 7](#_Toc3668)

[4.2.1 接口定义使用的参数列表 7](#_Toc23191)

[4.2.2 接口列表 8](#_Toc5273)

[4.3 补充说明 9](#_Toc23585)

[5 模块设计 9](#_Toc28434)

[5.1 StatusArray 9](#_Toc23314)

[5.1.1 功能 9](#_Toc17479)

[5.1.2 整体框图 9](#_Toc28149)

[5.1.3 接口列表 9](#_Toc14902)

[5.1.4 接口时序 11](#_Toc19972)

[5.1.5 关键电路 11](#_Toc27545)

[5.2 SelectPolicy 11](#_Toc32721)

[5.2.1 功能 11](#_Toc21616)

[5.2.2 整体框图 12](#_Toc23086)

[5.2.3 接口列表 12](#_Toc22484)

[5.3 DataArray 13](#_Toc9521)

[5.3.1 功能 13](#_Toc20826)

[5.3.2 整体框图 13](#_Toc6800)

[5.3.3 接口列表 13](#_Toc30350)

[5.4 PayloadArray 14](#_Toc6567)

[5.4.1 功能 14](#_Toc12624)

[5.4.2 整体框图 15](#_Toc13543)

[5.4.3 接口列表 15](#_Toc8733)

[5.5 OldestSelection 16](#_Toc11133)

[5.5.1 功能 16](#_Toc12674)

[5.5.2 整体框图 16](#_Toc14708)

[5.5.3 接口列表 16](#_Toc4687)

[5.5.4 补充说明 17](#_Toc14748)

[5.6 WakeupQueue 17](#_Toc25159)

[5.6.1 功能 17](#_Toc12287)

[5.6.2 整体框图 17](#_Toc17756)

[5.6.3 接口列表 17](#_Toc18604)

[5.6.4 补充说明 18](#_Toc30645)

[5.7 DataSelect 18](#_Toc30236)

[5.7.1 功能 18](#_Toc29976)

[5.7.2 整体框图 19](#_Toc29565)

[5.7.3 接口列表 19](#_Toc2158)

[5.7.4 补充说明 19](#_Toc21165)

[5.8 BypassNetwork 20](#_Toc27757)

[5.8.1 功能 20](#_Toc15014)

[5.8.2 整体框图 20](#_Toc11964)

[5.8.3 接口列表 20](#_Toc25256)

[5.8.4 补充说明 21](#_Toc20745)

[5.9 各类 ImmExtractor 22](#_Toc30572)

[5.9.1 功能 22](#_Toc16353)

[5.9.2 整体框图 24](#_Toc17178)

[5.9.3 接口列表 24](#_Toc13652)

[5.9.4 补充说明 25](#_Toc30181)

[5.10 AgeDetector 25](#_Toc3496)

[5.10.1 功能 25](#_Toc7276)

[5.10.2 整体框图 27](#_Toc1701)

[5.10.3 接口列表 27](#_Toc31325)

[5.10.4 补充说明 27](#_Toc26176)

1. 简介
   1. 文档介绍

本文当是南湖v2的AS文档，描述Reservation Station架构设计。

本文档主要用于指导芯片模块的详细设计及验证。

* 1. 参考文档
  2. 术语说明

表1.1 术语说明

|  |  |  |
| --- | --- | --- |
| **缩写** | **全称** | **描述** |
| CRU | Clock Reset Unit | 时钟复位单元 |
| OH | One Hot | 独热码。仅有唯一bit为1的编码方式。 |
| FU | Function Unit | 功能单元，运算的基本算子 |
| RS | Reservation Station | 保留站。在功能单元FU前面的uop缓存。 |
|  |  |  |

* 1. 技术背景

1. 设计规格
2. 功能描述
   1. 功能概述

RS 模块暂存来自 Dispatch 模块的指令信息，暂存指令使用的源操作数。RS 模块收集 FU 的写回信号、RS 发出的快速唤醒信号，随时更新 RS 内指令的源寄存器状态。在指令使用的源寄存器准备好且 FU 空闲时，RS 模块会根据 SelectPolicy 中的指令选择策略将指令和源操作数发送给相应的 FU。

* 1. 功能详述
     1. 接收并暂存来自 Dispatch 的指令信息

RS 模块通过 fromDispatch 端口接收来自 Dispatch 的指令后，将源寄存器状态信息通过 StatusArray 的 update 端口保存在 StatusArray 中，将指令信息通过 PayloadArray 的 write 端口保存到 PayloadArray 中。

具体存入的 entry 由 SelectPolicy 模块指定。SelectPolicy 依据当前 RS 中的空 entry 情况选出1-4个空位(JumpRS 1个空位，MduRS4个空位，其余是两个)，传给 StatusArray 和 PayloadArray，作为存入指令信息的 entry。

* + 1. 维护指令源物理寄存器状态

RS 中指令源物理寄存器初始状态，为指令经过 Dispatch2Rs 模块时查询 BusyTable 的结果，该状态会在 StatusArray 中随着执行单元写回或其它 RS 快速唤醒而更新。当有执行单元写回了源物理寄存器，写回信号通过 RS 的 slowports 端口到 StatusArray wakeup 端口，更新源寄存器的状态为 ready。当有 RS 发出了快速唤醒信号，该信号通过 RS 的 fastUopIn 端口也到 StatusArray wakeup 端口，更新源寄存器的状态为 ready。

* + - 1. 写回唤醒和快速唤醒

RS 中基础的唤醒机制是写回唤醒，即某 FU 写回信号的目的物理寄存器是该指令的源物理寄存器，当写回信号传递到 RS 时，即可更新寄存器状态。

有些指令有确定执行周期数，且配置了最高的写回优先级。在这些指令离开 RS 时，就已经可以准确估计它们执行完成并写回的时间。快速唤醒机制就是基于这样的原理实现的。当可以具备快速唤醒能力的指令将离开 RS 时，RS 中的 WakeupQueue 就入队该指令信息，延迟 n 个周期后出队，n 是指令的固定执行周期数。在该指令执行完成前一拍，WakeupQueue 出队该指令，将该指令的目的物理寄存器信息发送到其它 RS 中，提前唤醒其它指令。下一拍，该指令的执行结果也以最高优先级通过仲裁器由 slowports 发送到 RS 中。

对于 Load 指令，由于访存流水线有多个执行周期，执行结果在流水线结尾处才能获取到，而在得到结果的前一拍，就能够知道这条访存指令能否命中tlb、命中dcache并且顺利执行。于是 LoadUnit 也会发出快速唤醒信号到 RS 中，对应 otherFastWakeup 。



图 1 otherFastWakeup 时序图

* + 1. 选择指令发射执行

对源操作数已经就绪的指令，可以按照 SelectPolicy 和 AgeDetector 两个模块的选择策略，选出发送到 FU 的指令。SelectPolicy 依据 RS 中已经占据的 entry 合并当拍进入 RS 的 entry，根据配置的不同选择策略，选出源操作数就绪的指令。SelectPolicy 支持两种入队策略和三种出队策略，在 SelectPolicy 中详细描述。AgeDetector 则依据指令的出队入队情况通过矩阵来维护指令的“age”，并选择出最“老”的一条指令。

* + 1. 访存指令重发

load与store指令从保留站发射出来后，保留站不会立马清空对应的指令，而是继续等待MemBlock发来反馈。MemBlock中的流水线会根据具体的执行情况判断是否需要让保留站重发指令。

load与store流水线是非阻塞的，即不能因为某些特殊情况将指令卡在某一级流水线。例如load指令流水线访问TLB进行地址转换时，发现TLB miss，不能将这条指令一直停留在这一级流水线中等待TLB命中，需要让它从流水线中退出，让后续拥有执行条件的指令继续运行。对于TLB miss的load指令，它本身是没有完成的，所以需要让保留站重新发射这条指令。

下图中，robIdx 为 0B、12、15、19、1c 的 load 指令都进行了重发。蓝线是来自 LoadUnit 的 feedbackFast 信号，重发原因为0b011，在 0 号出队端口重发。

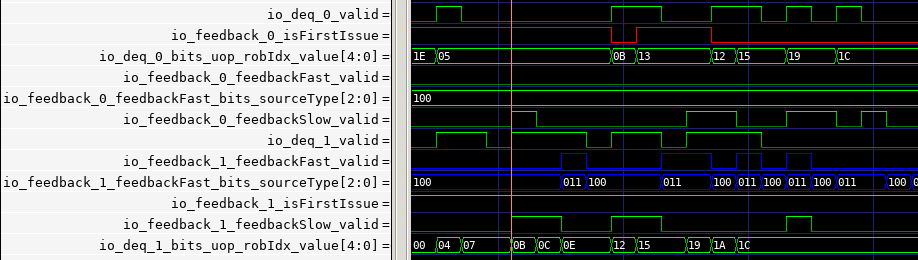


图 2 访存指令重发

1. 总体设计
   1. 整体框图

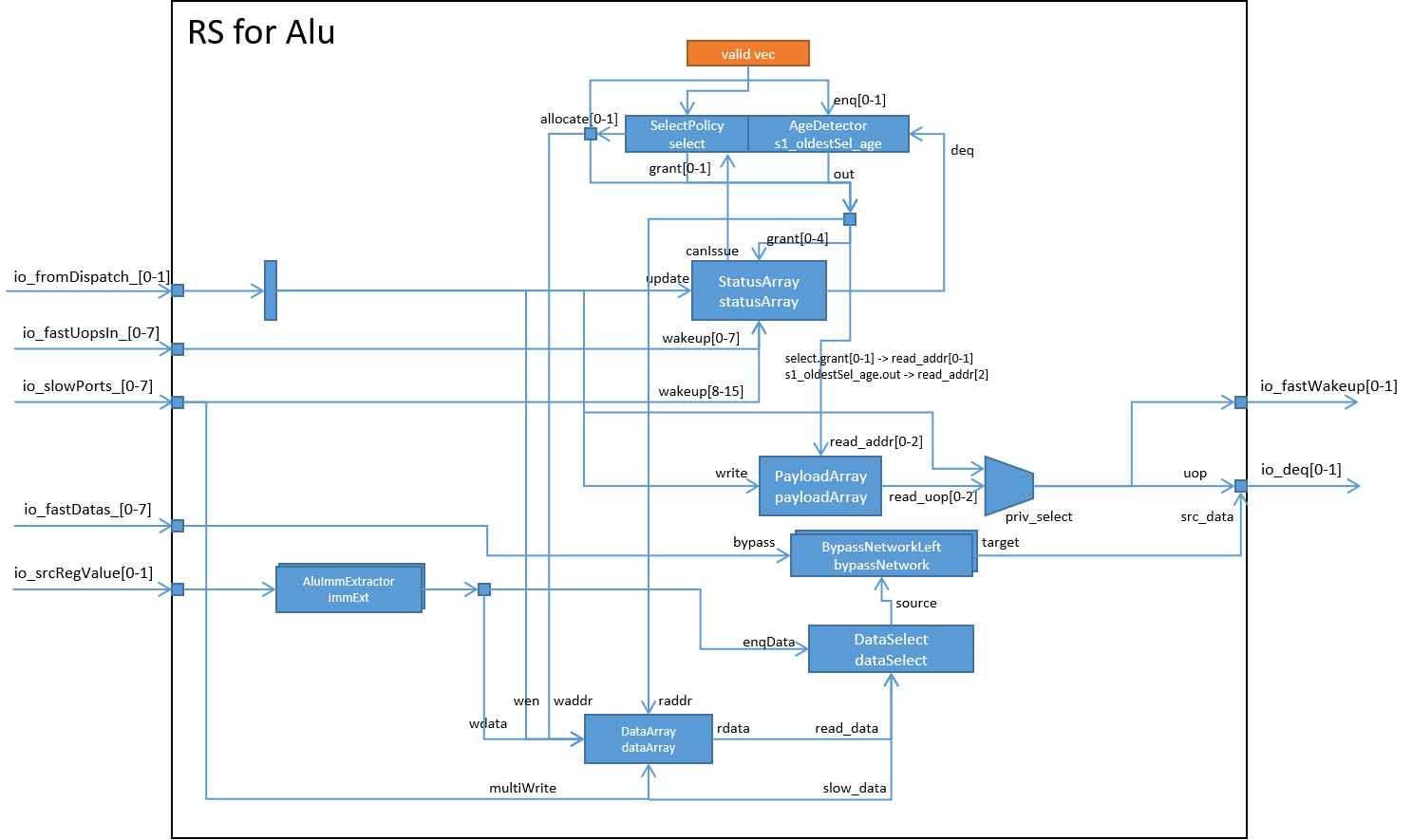


图4.1 AluRS模块整体框图

注：priv\_select优先级如下

deq[0]: payload.read[2] > payload.read[0] > uop\_from\_dispatch[1]

deq[1]: payload.read[1] > uop\_from\_dispatch[0]

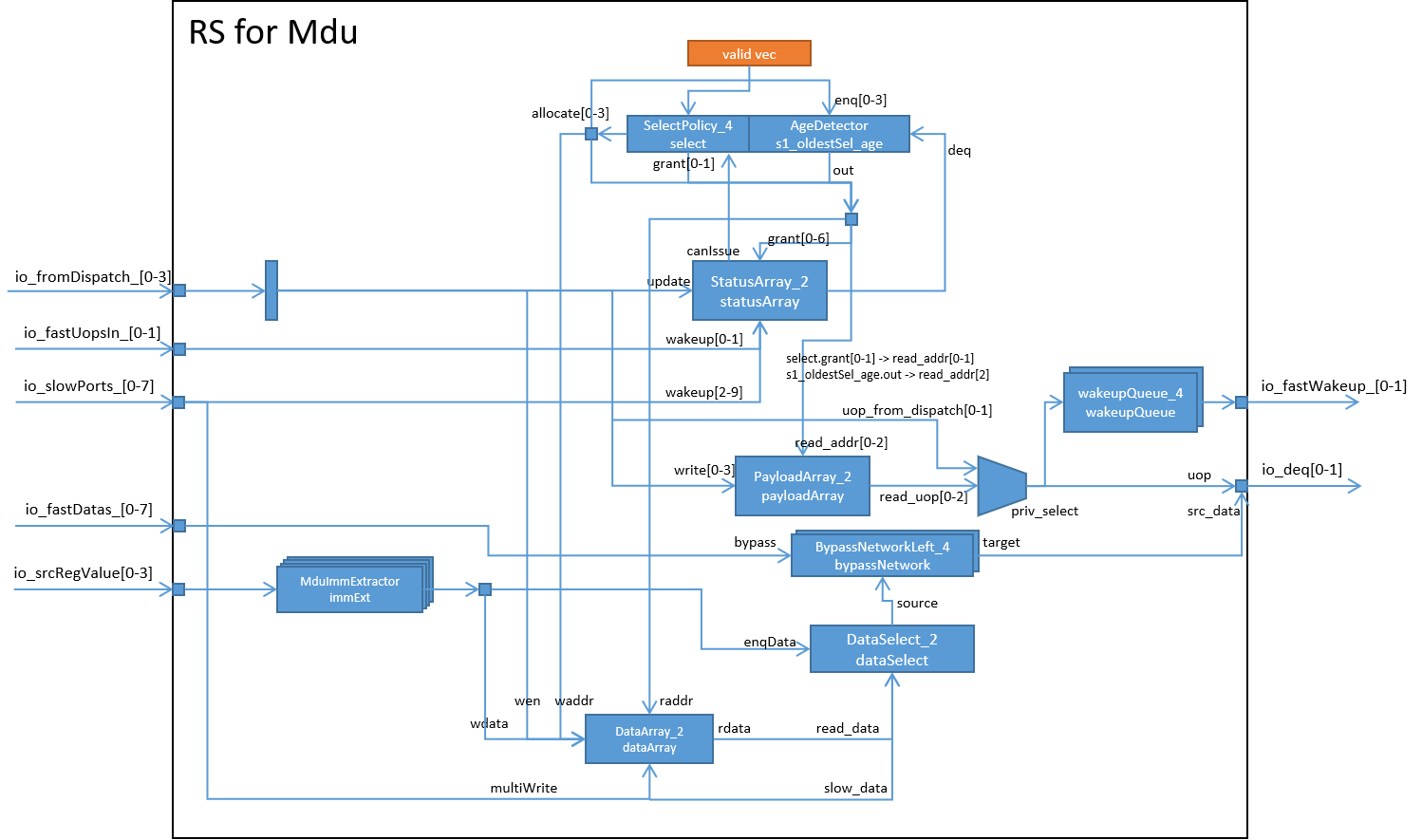


图4.2 MduRS模块整体框图

注：priv\_select优先级如下

deq[0]: payload.read[2] > payload.read[0] > uop\_from\_dispatch[1]

deq[1]: payload.read[1] > uop\_from\_dispatch[0]

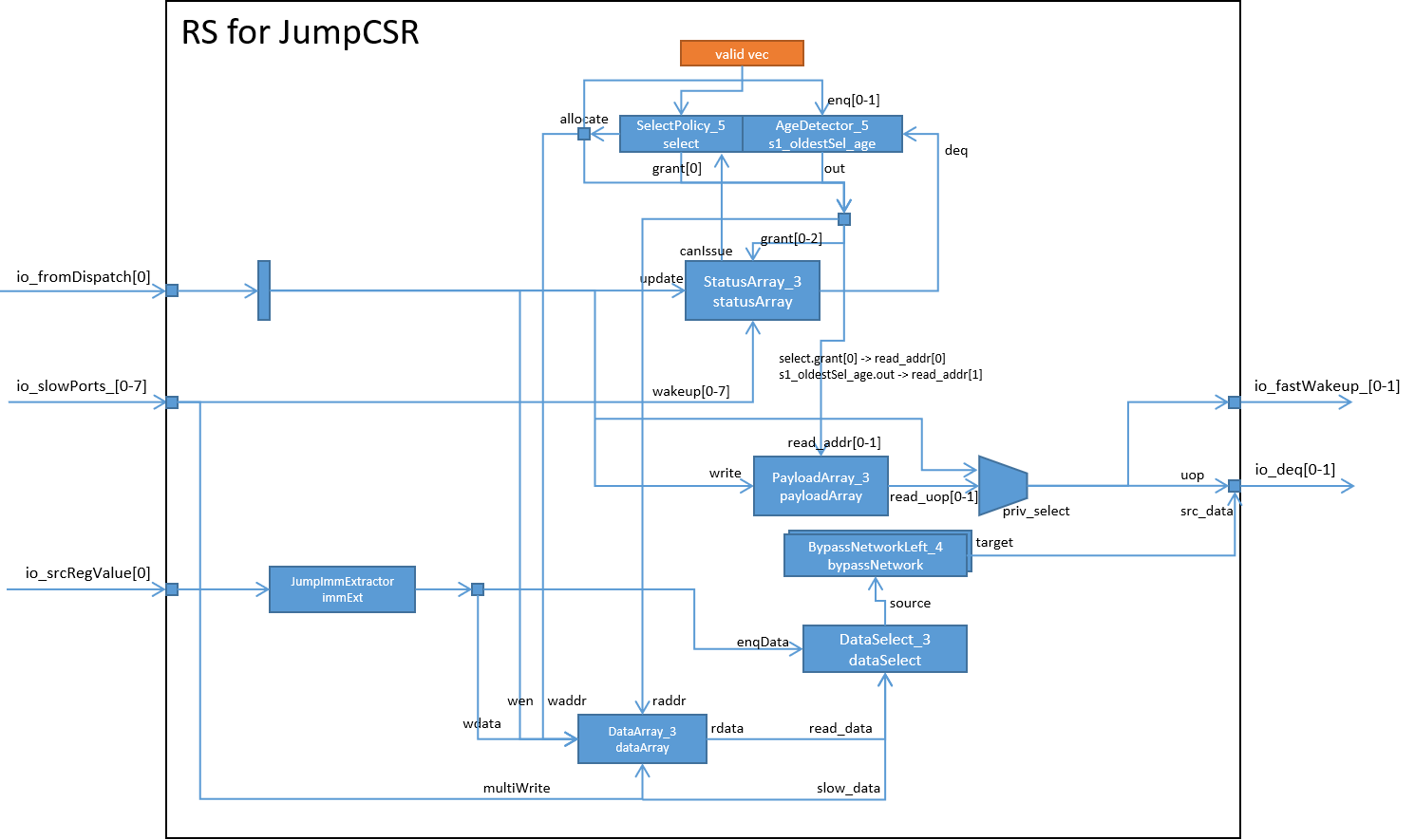


图4.3 JumpRS模块整体框图

注：priv\_select优先级如下

deq: payload.read[1] > payload.read[0] > uop\_from\_dispatch[0]

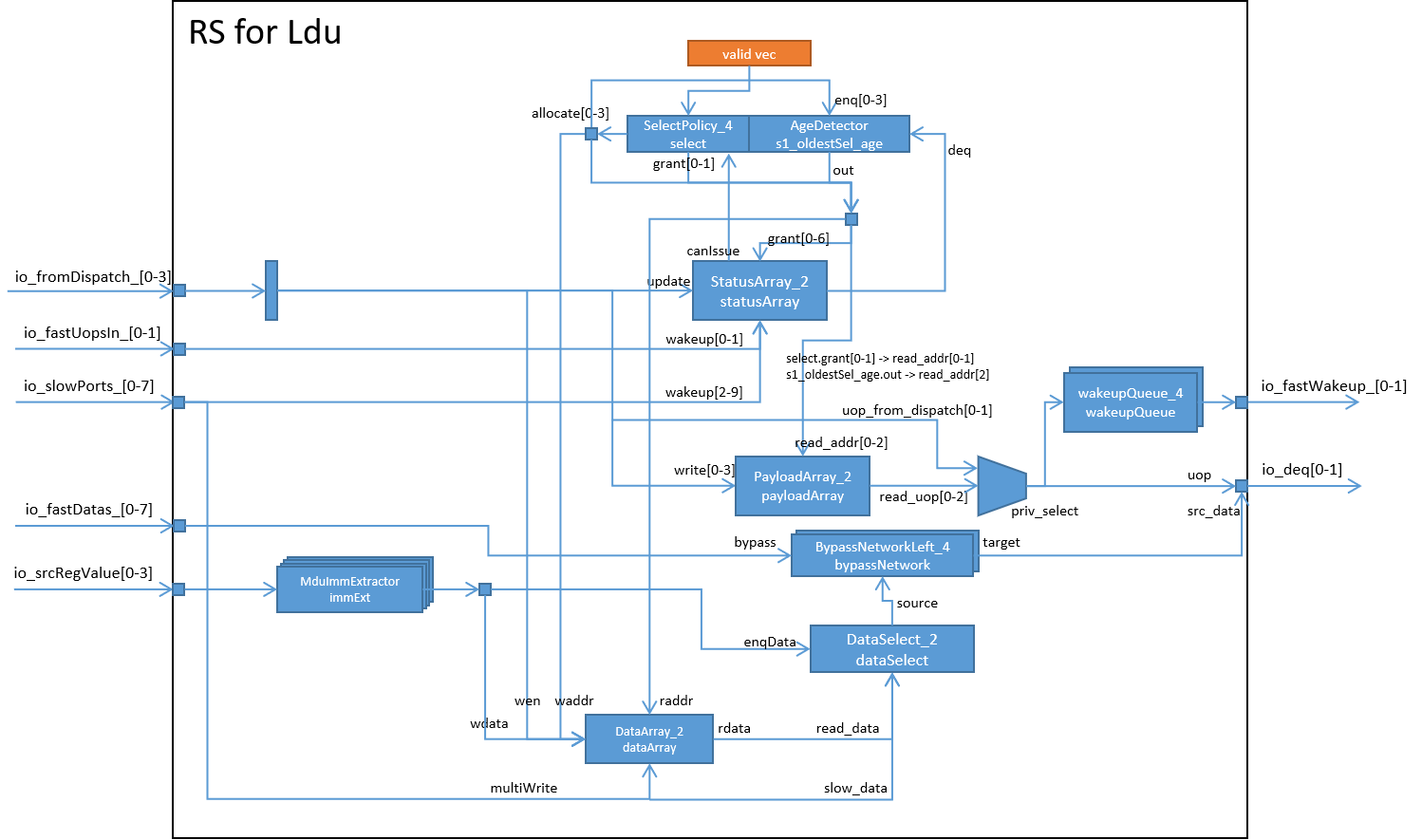


图4.3 LduRS模块整体框图

注：priv\_select优先级如下

deq[0]: payload.read[2] > payload.read[0] > uop\_from\_dispatch[1]

deq[1]: payload.read[1] > uop\_from\_dispatch[0]

* 1. 接口列表
     1. 接口定义使用的参数列表

表4.1 Reservation Station参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| numEntries | RS模块配置的entry数量 | 随不同RS模块的配置而变化 |
| numEnq | RS模块配置的单周期可入队的uop数量 | 随不同RS模块的配置而变化 |
| numDeq | RS模块配置的单周期可出队的uop数量 | 随不同RS模块的配置而变化，不超过 2 |
| numFastWakeup | 快速唤醒端口数量 | 随不同RS模块的配置而变化 |
| numWakeup | 写回唤醒端口数量 | 与寄存器堆写端口数量相同，整型和浮点都是8个 |
| VAddrBits | 虚拟地址位宽 | 恒为39 |
| lduCnt | 访存单元个数 | 2 |

* + 1. 接口列表

表4.2 Reservation Station 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| io\_redirect | I | ROB | RS | Redirect类型 | 来自ROB的重定向信号，详见《重要Bundle描述》的Redirect。 |
| io\_fromDispatch\_X | I | Dispatch2Rs | RS | MicroOp类型 | 来自dispatch的uop入队信号。详见《重要Bundle描述》的MicroOp。  X的取指范围为[0, numEnq)。 |
| io\_srcRegValue\_X\_Y | I | Dispatch2Rs | RS | 64 | 第 X 个入队uop的第 Y 个源操作数。由Dispatch2Rs从物理寄存器堆中读取。 |
| io\_deq\_X | O | RS | FUBlock | MicroOp类型 | 从保留站出队的指令信息，送往FU。详见《重要Bundle描述》的MicroOp。 |
| io\_fastUopsIn\_X | I | 当前RS和其它RS  MemBlock | RS | MicroOp类型 | 来自RS或MemBlock的快速唤醒信号。详见《重要Bundle描述》的MicroOp。 |
| io\_fastDatas\_X | I | WbAribter | RS | 64 | 与fastUopsIn对应的旁路数据。  X的取指范围[0,numFastWakeup) |
| io\_slowPorts\_X | I | WbAribter | RS | ExuOutput类型 | 经过仲裁的写回数据。  X的取指范围[0,numWakeup)  Bundle详见《重要Bundle描述》的ExuOutput。 |
| io\_fastWakeup\_X | O | RS | 当前RS和其它RS | MicroOp类型 | 固定周期数且优先写回指令发射的伴随快速唤醒信号，比写回提信号前一拍到达RS。  对固定周期指令对应的RS，X的取指范围[0, numDeq)，其它RS为0。  Bundle详见《重要Bundle描述》的MicroOp。 |
| io\_jump\_jumpPc | I | CtrlBlock | RS | VAddrBits | 这条跳转指令的所在的pc |
| io\_jump\_jalr\_target | I | CtrlBlock | RS | VAddrBits | jalr指令的跳转目标 |
| io\_feedback | I/O | MemBlock | RS | MemRSFeedbackIO类型 | X 的取值范围{0, 1}  对LoadRS，信号来自LoadUnit；对StoreAddrRS和StoreDataRs，信号来自StoreUnit。对 RS 内指令的重发反馈信号。详见《重要Bundle描述》的MemRSFeedbackIO。 |
| io\_checkwait\_stIssuePtr | I | MemBlock | RS | SqPtr类型 | TODO |
| io\_checkwait\_memWaitUpdateReq | I | MemBlock | RS | MemWaitUpdateReq类型 | 详见《重要Bundle描述》的MemWaitUpdateReq。 |
| io\_load\_X\_fastMatch | O | RS | LoadUnit | lduCnt | loadRS中，来自第X个LoadUnit的目的寄存器和该指令的源寄存器匹配。  X 的取值范围[0, lduCnt)，下同。 |
| io\_load\_X\_fastImm | O | RS | LoadUnit | lduCnt | 匹配指令对应的指令立即数 |
| io\_fmaMid | I/O | fmac | RS | FMAMidResultIO类型 | 用于保存fmac中间结果的信号组 |

* 1. 补充说明

1. 模块设计
   1. StatusArray
      1. 功能

StatusArray中保存当前RS持有的uop对应的源寄存器等状态，StatusArray会监听其它功能单元执行结果，接受uop实际的issue状态，以此更新StatusArray中的寄存器。StatusArray将能够issue的uop传递到RS，供RS参考调度使用。

* + 1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| numEntries | RS模块配置的entry数量 | 随不同RS模块的配置而变化 |
| numEnq | RS模块配置的单周期可入队的uop数量 | 随不同RS模块的配置而变化 |
| numDeq | RS模块配置的单周期可出队的uop数量 | 随不同RS模块的配置而变化，不超过 2 |
| numFastWakeup | 快速唤醒端口数量 | 随不同RS模块的配置而变化 |
| numWakeup | 写回唤醒端口数量 | 与寄存器堆写端口数量相同，整型和浮点都是8个 |
| VAddrBits | 虚拟地址位宽 | 恒为39 |
| lduCnt | 访存单元个数 | 2 |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| io\_redirect |  |  |  |  | 来自ROB的冲刷信号 |
| \_valid | I | ROB | StatusArray | 1 | 冲刷信号有效 |
| \_bits\_robIdx\_flag | I | ROB | StatusArray | 1 |  |
| \_bits\_robIdx\_value | I | ROB | StatusArray | 8 | 与flag信号组成rob循环队列指针 |
| \_bits\_level | I | ROB | StatusArray | 1 | 是否冲刷这个指针对应的指令信息  0：冲刷  1：不冲刷 |
|  |  |  |  |  |  |
| io\_isValid | O | StatusArray | RS | numEntries | 同下，但是延迟一个周期 |
| io\_isValidNext | O | StatusArray | SelectPolicy | numEntries | entry有效向量，根据update信号、deq信号与当前valid信号得出的当拍更新信号 |
| io\_canIssue | O | StatusArray | SelectPolicy | numEntries | 同下，但是延迟一个周期 |
| io\_canIssueNext | O | StatusArray | SelectPolicy | numEntries | RS中对应entry的uop可以参与issue，根据当拍update信号得出的当拍信号 |
| io\_flushed | O | StatusArray | AgeDetector | numEntries | 该entry有效且收到冲刷信号的下一拍，或该entry成功发射。用于更新entry的新旧状态 |
|  |  |  |  |  |  |
| io\_update\_X | I | Dispatch | StatusArray | - | 更新StatusArray内部状态的信号，X随更新信号来源数量而定 |
| \_enable | I | Dispatch | StatusArray | 1 | 该update信号有效 |
| \_addr | I | Dispatch | StatusArray | numEntries | 更新第addr的status |
| \_data\_srcState\_Y | I | Dispatch | StatusArray | 1 | 第Y个源操作数的状态 |
| \_data\_psrc\_Y | I | Dispatch | StatusArray | 8 | 第Y个源寄存器的状态 |
| \_data\_robIdx\_flag | I | Dispatch | StatusArray | 1 | - |
| \_data\_robIdx\_value | I | Dispatch | StatusArray | 8 | 与addr位置status关联的rob指针 |
|  |  |  |  |  |  |
| io\_wakeup\_X |  | FU或RS | StatusArray |  | 唤醒信号，X=allWakeUp |
| \_valid | I | FU或RS | StatusArray | 1 | 唤醒信号有效 |
| \_bits\_ctrl\_rfWen | I | FU或RS | StatusArray | 1 | 唤醒信号对应的物理寄存器的写信号 |
| \_bits\_pdest | I | FU或RS | StatusArray | 8 | 唤醒信号对应的物理寄存器 |
|  |  |  |  |  |  |
| io\_wakeupMatch\_X\_Y | O | StatusArray | RS | allWakeUp | X in [0, numEntries)  Y in [0, numSrc)  唤醒信号命中  io\_wakeupMatch\_X\_Y[i]表示第i个wakeup信号的目的寄存器与第X个entry的第Y个源寄存器匹配。用于唤醒RS中源寄存器相同的指令 |
|  |  |  |  |  |  |
| io\_deqResp\_X |  |  |  |  | 如果是访存指令的RS，则信号来自MemBlock，否则来自RS。 |
| \_valid | I | RS或MemBlock | StatusArray | 1 | 出队信号有效 |
| \_bits\_rsMask | I | RS或MemBlock | StatusArray | numEntries | 出队onehot向量 |
| \_bits\_success | I | RS或MemBlock | StatusArray | 1 | 指令发射成功 |

* + 1. 接口时序
    2. 关键电路
  1. SelectPolicy
     1. 功能

SelectPolicy模块有以下两个功能。

选出numEnq个空白项

选出numDeq的有效项

SelectPolicy模块根据当前输入的validVec，每个周期选出空白项（validVec置0的项），同时每个周期选出有效项（validVec置1的项）。

每个SelectPolicy根据配置不同，有不同的选择策略。

SelectPolicy使用的选择空白项策略如下。

|  |  |  |
| --- | --- | --- |
| **numEnq** | **选择策略** | **选择策略描述** |
| 1 | circ | 从向量最前和最后的空白项交替选取 |
| 2 | center | 从向量中间向两边各选取一个 |
| >2 | oddeven | 将向量按序号奇偶分成两个部分，分别应用circ选取策略 |

SelectPolicy使用的选择有效项策略如下。

|  |  |  |
| --- | --- | --- |
| **numDeq** | **选择策略** | **选择策略描述** |
| 1 | naive | 从前到后按顺序选取 |
| >=2 | circ | 从向量最前和最后的空白项交替选取 |
| numEnq>2且numEntries>32 | oddeven | 将向量按序号奇偶分成两个部分，分别应用circ选取策略 |

* + 1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| numEntries | RS模块配置的entry数量 | 随不同RS模块的配置而变化 |
| numEnq | RS模块配置的单周期可入队的uop数量 | 随不同RS模块的配置而变化 |
| numDeq | RS模块配置的单周期可出队的uop数量 | 随不同RS模块的配置而变化，不超过 2 |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| clock | I | RS | SelectPolicy | 1 |  |
| reset | I | RS | SelectPolicy | 1 |  |
| io\_validVec | I | RS | SelectPolicy | numEntries | entry有效信号向量 |
| io\_allocate\_X\_bits | O | RS | SelectPolicy | numEntries | 选取的entry空闲向量，置1时validVec对应位置必须是0。  可以视作RS将要入队的向量。  X的取值在[0, numEnq)之间。 |
| io\_request | I | RS | SelectPolicy | numEntries |  |
| io\_grant\_X |  |  |  |  | RS中将要出队的向量信号。  X取值在[0, numDeq)之间 |
| \_valid | O | SelectPolicy | RS | 1 | 出队向量有效 |
| \_bits | O | SelectPolicy | RS | numEntries | 出队OH向量 |
| io\_balance |  |  |  |  | 当RS配置需要平衡输出且出口处FU有两个时，才有此接口 |
| \_tick | I | RS | SelectPolicy | 1 | 每轮出队信号有效时，tick置1 |
| \_out | O | SelectPolicy | RS | 1 | out置1时，表明出队的两个uop与实际应出队的顺序相反。如果应出队的仅有一个uop，那么io\_grant\_0将无效，而不是io\_grant\_1。 |

* 1. DataArray
     1. 功能

DataArray模块保存当前RS中uop使用的源操作数。当RS入队时，将已准备好的源操作数写入DataArray中。DataArray将监听其他FU产出的数据结果，并将源操作数更新。当RS出队时，DataArray将输出uop需要的源操作数供FU使用。

部分FU会将中间运算结果通过部分写（partialWrite）接口保存在DataArray中。DataArray支持对中间结果的旁路加速读取，即同一周期同时部分写并读出同一个entry的数据时，将会把部分写的数据作为读端口的输出。

* + 1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| XLEN | 香山处理器支持数据运算最大位宽 | 恒等于64 |
| numSrc | RS模块中uop需要的源操作数个数 | 根据RS后连接的不同FU需要的源操作数不同，numSrc可以为1，2，3 |
| numEntries | RS模块配置的entry数量 | 随不同RS模块的配置而变化，16~32之间 |
| numEnq | RS模块配置的单周期可接收的uop数量 | 1，2 |
| numDeq | RS模块配置的单周期可送出的uop数量 | 1，2，3，4 |
| numWakeup | RS模块配置的监听FU单元运算结果数据的数量 | 随不同RS模块的配置而变化 |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| clock | I | RS | DataArray | 1 |  |
| io\_read\_X |  |  |  |  | uop从RS中出队时，用这个接口读取uop所用的源操作数。部分FU读取中间运算结果时，也会用到这个接口。  X的取值范围为[0, numDeq+1)。 |
| \_addr | I | RS | DataArray | numEntries | 要读取数据的OH向量 |
| \_data\_Y | O | DataArray | RS | XLEN | 读取数据的结果。  Y的取值范围为[0, numSrc)。 |
|  |  |  |  |  |  |
| io\_write\_X |  |  |  |  | uop进入RS时，用这个接口向DataArray写入uop所用的源操作数。  X的取值范围为[0, numEnq)。 |
| \_enable\_Y | I | RS | DataArray | 1 | 是否有该源操作数。如果uop不需要读取对应的源操作数，enable置0。  Y的取值范围为[0, numSrc)。 |
| \_mask\_Y | I | RS | DataArray | 1 | 是否需要写入数据。如果在uop进入RS时数据未准备好，mask置0。 |
| \_addr | I | RS | DataArray | numEntries | 要写入数据的OH向量 |
| \_data\_Y | I | RS | DataArray | XLEN | 要写入的数据 |
|  |  |  |  |  |  |
| io\_multiWrite\_X |  |  |  |  | RS模块配置的监听FU单元数据产出使用的接口。  X的取值范围为[0, numWakeup) |
| \_enable | I | RS | DataArray | 1 | 写数据有效 |
| \_addr\_Y | I | RS | DataArray | numEntries | 要写入数据的OH向量。  X的取值范围为[0, numSrc) |
| \_data | I | RS | DataArray | XLEN | 要写入的数据 |
|  |  |  |  |  |  |
| io\_delayedWrite\_X |  |  |  |  |  |
| \_mask\_Y | I | RS | DataArray | 1 |  |
| \_addr | I | RS | DataArray | numEntries |  |
| \_data\_Y | I | RS | DataArray | XLEN |  |
|  |  |  |  |  |  |
| io\_partialWrite\_X |  |  |  |  | 部分FU使用DataArray保存中间结果时，用到这个接口。  X的取值范围为[0, numDeq) |
| \_enable\_Y | I | RS | DataArray | 1 | 是否覆盖写该源操作数。  Y的取值范围为[0, numSrc)。 |
| \_addr | I | RS | DataArray | numEntries | 要写入数据的OH向量 |
| \_data\_Y | I | RS | DataArray | XLEN | 要写入的数据 |

* 1. PayloadArray
     1. 功能

PayloadArray模块用于保存RS中uop的指令信息。RS入队时写入，出队时读出（不擦除）。根据不同的RS配置，PayloadArray模块保存的uop信息不完全相同，但是不同配置的PayloadArray功能完全一致。

* + 1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| XLEN | 香山处理器支持数据运算最大位宽 | 恒等于64 |
| numSrc | RS模块中uop需要的源操作数个数 | 根据RS后连接的不同FU需要的源操作数不同，numSrc可以为1，2，3 |
| numEntries | RS模块配置的entry数量 | 随不同RS模块的配置而变化，16~32之间 |
| numEnq | RS模块配置的单周期可接收的uop数量 | 1：JumpCSR  2：Alu，Fmac  4：Mdu，Fmisc |
| numDeq | RS模块配置的单周期可送出的uop数量 | 1：JumpCSR  2：Alu，Fmac，Mdu，Fmisc |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| clock | I | RS | PayloadArray | 1 |  |
| io\_read\_X |  |  |  |  | uop从RS中出队时，用这个接口读取uop信息。  X的取值范围为[0, numDeq+1)。 |
| \_addr | I | RS | PayloadArray | numEntries | 要读取数据的OH向量 |
| \_data\_\* | O | PayloadArray |  | - | 读取的uop信息。  Y的取值范围为[0, numSrc)。 |
|  |  |  |  |  |  |
| io\_write\_X |  |  |  |  | uop进入RS时，用这个接口向DataArray写入uop所用的源操作数。  X的取值范围为[0, numEnq)。 |
| \_enable | I |  | PayloadArray | 1 | 写信号有效 |
| \_addr | I |  | PayloadArray | numEntries | 要写入数据的OH向量 |
| \_data\_\* | I |  | PayloadArray | - | 写入的uop信息 |

* 1. OldestSelection
     1. 功能

OldestSelection模块用于检查SelectPolicy选择的RS出队uop是否包含最早进入RS的uop。

* + 1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| XLEN | 香山处理器支持数据运算最大位宽 | 恒等于64 |
| numSrc | RS模块中uop需要的源操作数个数 | 根据RS后连接的不同FU需要的源操作数不同，numSrc可以为1，2，3 |
| numEntries | RS模块配置的entry数量 | 随不同RS模块的配置而变化 |
| numEnq | RS模块配置的单周期可接收的uop数量 |  |
| numDeq | RS模块配置的单周期可送出的uop数量 |  |
| numWakeup | RS模块配置的监听FU单元运算结果数据的数量 | 随不同RS模块的配置而变化 |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| io\_in\_X\_valid | I | 1 | RS | OldestSelection | SelectPolicy选择的出队OH向量有效  X的取值范围为[0, numDeq)。 |
| io\_in\_X\_bits | I | numEntries | RS | OldestSelection | SelectPolicy选择的第X个出队OH向量 |
| io\_oldest\_valid | I | 1 | RS | OldestSelection | RS如果不为空，则为1，即存在最早入队的uop |
| io\_oldest\_bits | I | numEntries | RS | OldestSelection | RS最早入队的uop对应OH向量 |
| io\_canOverride | I | numDeq | RS | OldestSelection | OH，第n位为1表示第n个出队向量可以被最早入队向量替换 |
| io\_isOverrided\_X | O | 1 | OldestSelection | RS | 在可以canOverride向量中选出的不是最早入队的向量。满足条件时对应位置1。 |

* + 1. 补充说明

OldestSelection模块实际生成的verilog代码中，部分信号在某些参数配置下被优化，但功能未改变。

* 1. WakeupQueue
     1. 功能

WakeupQueue模块是个FIFO，在uop入队后延迟一定的周期数出队。如果在RS里配置了fixedLatency参数，则会实例化numDeq个WakeupQueue。uop从RS出队送进FU执行时，也同时会被送入WakeupQueue。当uop执行完毕，WakeupQueue的延迟出队信号也同时有效，用于唤醒其它uop参与调度执行。

当重定向信号有效时，将会冲刷晚于ROB指针（是否等于由level信号指定）进入ROB的uop。

* + 1. 整体框图
    2. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| clock | I |  | WakeupQueue | 1 |  |
| reset | I |  | WakeupQueue | 1 |  |
|  |  |  |  |  |  |
| io\_in |  |  |  |  | 使用RS入队的uop信号作为WakeupQueue的入队信号 |
| \_valid | I | RS | WakeupQueue | 1 | WakeupQueue的入队信号有效 |
| \_bits\_ctrl\_rfWen | I | RS | WakeupQueue | 1 |  |
| \_bits\_pdest | I | RS | WakeupQueue | 8 |  |
| \_bits\_robIdx\_flag | I | RS | WakeupQueue | 1 |  |
| \_bits\_robIdx\_value | I | RS | WakeupQueue | 8 |  |
|  |  |  |  |  |  |
| io\_out |  |  |  |  | WakeupQueue的出队信号，用于唤醒相应的uop |
| \_valid | O | WakeupQueue | RS | 1 | 出队信号有效 |
| \_bits\_ctrl\_rfWen | O | WakeupQueue | RS | 1 |  |
| \_bits\_pdest | O | WakeupQueue | RS | 8 |  |
|  |  |  |  |  |  |
| io\_redirect |  |  |  |  | 用于冲刷WakeupQueue的重定向信号 |
| \_valid | I | RS | WakeupQueue | 1 |  |
| \_bits\_robIdx\_flag | I | RS | WakeupQueue | 1 |  |
| \_bits\_robIdx\_value | I | RS | WakeupQueue | 8 |  |
| \_bits\_level | I | RS | WakeupQueue | 1 |  |
|  |  |  |  |  |  |

* + 1. 补充说明

WakeupQueue仅在固定执行周期的FU对应的RS中实例化。

* 1. DataSelect
     1. 功能

DataSelect模块用于为出队uop选择对应的源操作数。源操作数的来源包括RS入队uop携带的源操作数，由SelectPolicy模块选择的uop在DataArray中对应的源操作数，由OldestSelect模块选择的最早进入RS的uop对应的源操作数和监听其它功能单元的输出四类。选出的uop对应的源操作数在readData信号组中，其它功能单元的输出数据在slowData信号组中，RS入队uop携带的源操作数则在enqData信号组中。

DataSelect模块选择数据的逻辑如下。

1.当旁路信号有效时，即刚入队RS的uop被选中出队时，

1.1如果监听的功能单元有数据产生，选择监听的功能单元产生的数据。

1.2否则，选择入队uop携带的源操作数。

2.否则，当强制选择最早入队的uop出队时，选择最早入队的uop的源操作数。

3.否则，选择SelectPolicy模块选出的uop对应的源操作数。

* + 1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| XLEN | 香山处理器支持数据运算最大位宽 | 恒等于64 |
| numSrc | RS模块中uop需要的源操作数个数 | 根据RS后连接的不同FU需要的源操作数不同，numSrc可以为1，2，3 |
| numEntries | RS模块配置的entry数量 | 随不同RS模块的配置而变化 |
| numEnq | RS模块配置的单周期可接收的uop数量 | 1，2 |
| numDeq | RS模块配置的单周期可送出的uop数量 | 1，2，3，4 |
| numWakeup | RS模块配置的监听FU单元运算结果数据的数量 | 随不同RS模块的配置而变化 |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| io\_doOverride\_X | I | RS | DataSelect | 1 | X的取值范围为[0, numDeq) |
| io\_readData\_X\_Y | I | DataArray | DataSelect | XLEN | 从DataArray读出的数据。  X的取值范围为[0, numDeq + 1)  Y的取值范围为[0, numSrc) |
| io\_fromSlowPorts\_X\_Y | I | RS | DataSelect | numWakeup | 监听功能单元的输出信号，1表示相应功能单元有输出结果。  X的取值范围为[0, numDeq + 1 + numEnq)。  Y的取值范围为[0, numSrc)。 |
| io\_slowData\_X | I | RS | DataSelect | XLEN | 功能单元的输出数据。  X的取值范围为[0,numWakeup)。 |
| io\_enqBypass\_X\_X | I | RS | DataSelect | 1 | 刚进入RS的uop被选中issue。 |
| io\_enqData\_X\_Y\_bits | I | RS | DataSelect | XLEN | 进入RS的第X个uop携带的第Y个源操作数数据。 |
| io\_deqData\_X\_Y | O | DataSelect | RS | XLEN | RS出队uop的源操作数。 |

* + 1. 补充说明

无。

* 1. BypassNetwork
     1. 功能

RS中 uop 入队对应的项会在 T 时刻更新到 StatusArray 和 PayloadArray 模块，而寄存器堆的读数据会在T+1时刻到达。如果T+1时刻被选择的指令是T时刻入队的指令，该指令的数据会经过 BypassNetwork 模块，与fast bypass的数据按一定规则合并后，再从 RS 输出。

BypassNetwork 按需要有两个不同的实例化模块，即 BypassNetworkLeft 和 BypassNetworkRight。BypassNetworkLeft 模块先对不同输入的bypass数据做选通，再保存到寄存器输出。BypassNetworkRight模块则先全部将bypass信号保存到寄存器中，再选通输出。后者使用以空间换时序的方式，将选通信号后移，缓解从执行单元的输出端到到旁路网络的输入端的时序压力。

当输入保持信号(io\_hold)有效时，BypassNetwork 模块保持 T 时刻的数据到 T+1 时刻，忽略bypass信号组的输入信号。

* + 1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| XLEN | 香山处理器支持数据运算最大位宽 | 恒等于64 |
| numWays | BypassNetwork 模块的数据通道数，即 RS 模块中 uop 需要的源操作数个数 | 根据 RS 后连接的不同FU需要的源操作数不同，numWays 可以为1，2，3 |
| numBypass | BypassNetwork 模块的每个数据通道的数据来源个数 | 与快速唤醒(fast wake up)的数量配置相同 |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| clock | I |  |  | BypassNetwork |  |
| io\_hold | I | 1 | RS | BypassNetwork | 是否将T时刻的数据保持到T+1时刻。 |
| io\_source\_X | I | XLEN | RS | BypassNetwork | 通过BypassNetwork模块的读寄存器数据。  X的取值范围为[0, numWays)。 |
| io\_target\_X | O | XLEN | BypassNetwork | ExeUnit(AluExeUnit, MulDivExeUnit, JumpCSRExeUnit...),  MemBlock | 结合了读寄存器数据和旁路数据的输出结果。  X的取值范围为[0, numWays)。 |
| io\_bypass\_Y\_valid\_X | I | 1 | RS | BypassNetwork | 来自第 Y 个旁路输入的数据即是第 X 个源操作数。  X的取值范围为[0, numWays)。  Y的取值范围为[0, numBypass)。 |
| io\_bypass\_Y\_data | I | XLEN | ExeUnit(AluExeUnit, MulDivExeUnit, JumpCSRExeUnit...),  MemBlock | BypassNetwork | 来自第 Y 个旁路输入的数据。 |

* + 1. 补充说明

chisel代码中不同RS的optDeqFirstStage参数决定了将实例化的不同BypassNetwork模块，optDeqFirstStage参数取值则与ExuUnit的readFpRf参数取值相反，即所有读取浮点寄存器的RS的optDeqFirstStage参数置0，将实例化BypassNetworkRight模块。反之，所有不读取浮点寄存器的RS将实例化BypassNetworkLeft模块

* 1. 各类 ImmExtractor
     1. 功能

RS中 uop 入队对应的项会在 T 时刻更新到 StatusArray 和 PayloadArray 模块，而寄存器堆的读数据会在T+1时刻到达。如果T+1时刻被选择的指令是T时刻入队的指令，该指令的数据会经过 BypassNetwork 模块，与fast bypass的数据按一定规则合并后，再从 RS 输出。

BypassNetwork 按需要有两个不同的实例化模块，即 BypassNetworkLeft 和 BypassNetworkRight。BypassNetworkLeft 模块先对不同输入的bypass数据做选通，再保存到寄存器输出。BypassNetworkRight模块则先全部将bypass信号保存到寄存器中，再选通输出。后者使用以空间换时序的方式，将选通信号后移，缓解从执行单元的输出端到到旁路网络的输入端的时序压力。

当输入保持信号(io\_hold)有效时，BypassNetwork 模块保持 T 时刻的数据到 T+1 时刻，忽略bypass信号组的输入信号。

* + - 1. srcType 编码

|  |  |  |
| --- | --- | --- |
| **编码** | **枚举名称** | **含义** |
| 2'b00 | reg | 源操作数来自整型寄存器 |
| 2'b01 | pc | 第 0 个源操作数为 pc |
| 2'b01 | imm | 第 1 个源操作数为立即数拓展 |
| 2'b10 | fp | 源操作数来自浮点寄存器 |

* + - 1. selImm 编码

|  |  |  |
| --- | --- | --- |
| **编码** | **枚举名称** | **含义** |
| 4'b0000 | IMM\_S | 对应 Store 指令的立即数编码和拓展方式。  12位立即数使用符号拓展到32位。out = {20{imm[11]}, imm[11:0]} |
| 4'b0001 | IMM\_SB | 对应条件分支指令的立即数编码和拓展方式。12位立即数末位补一位0变为13位，再使用符号拓展到32位。  out = {19{imm[11]}, imm[11:0], 1'b0} |
| 4'b0010 | IMM\_U | 对应 PC 作为源操作数的指令立即数编码和拓展方式。  20位立即数低位补12位0拓展到32位。  out = {imm[19:0], 12'b0} |
| 4'b0011 | IMM\_UJ | 对应 jalr 指令立即数编码和拓展方式。  20位立即数低位补1位0后，符号拓展到32位。  out = {11{imm[19]}, imm[19:0], 1'b0} |
| 4'b0100 | IMM\_I | 对应需要 I 类型指令的立即数编码和拓展方式。  12位立即数符号拓展到32位。  out = {20{imm[11]}, imm[11:0]} |
| 4'b0101 | IMM\_Z | 对应需要CSR读写指令的立即数编码和拓展方式。  17位立即数（高5位为uimm，低12位为CSR寄存器编号）零拓展到32位。  out = {15'b0, imm[16:0]} |
| 4'b0110 | INVALID\_INSTR | 对应香山与NEMU模拟器联合仿真时的trap指令。 |
| 4'b0111 | IMM\_X | 对应不含立即数的指令。 |
| 4'b1000 | IMM\_B6 | 对应指令暂时未知，TODO。  6位立即数零拓展到32位。  out = {27'b0, imm[5:0]} |

* + - 1. 功能描述

ImmExtractor 模块位于 RS 的入口，用于对从 RS 入队的 uop 的做立即数拓展和源操作数选通。

ImmExtractor 有多个特化版本，具体功能如下所示。

|  |  |  |
| --- | --- | --- |
| **编码** | **枚举名称** | **含义** |
| AluImmExtractor | Alu 模块前的立即数拓展和选通模块 | 共2个源操作数  直通第 0 个源操作数  对第 1 个源操作数的srcType  非imm，直通  为imm，对selImm  为IMM\_U，用对应的拓展方式  非IMM\_U，用IMM\_I对应的拓展方式 |
| MduImmExtractor | Mdu 模块前的立即数拓展和选通模块 | 共2个源操作数  直通第 0 个源操作数  对第 1 个源操作数的srcType  非imm，直通  为imm，用IMM\_I对应的拓展方式 |
| JumpImmExtractor | Jump 模块前的立即数拓展和选通模块 | 共2个源操作数  此模块输入的立即数在jalr\_target信号中，该信号由预译码计算得到。  对第 0 个源操作数的srcType  非pc，直通  pc，将39位jump\_pc符号拓展到64位  对第 1 个源操作数的srcType  非imm，直通  imm，将39位jalr\_target零拓展到64位 |
| LoadImmExtractor | 用于拓展lui-load指令融合中被融合lui指令的20位立即数 | 共1个源操作数  对第 0 个源操作数的srcType  非imm，直通  imm，将{{psrc\_1, psrc\_0, ctrl\_imm}[31:12], 12'b0}符号拓展到64位 |
| ImmExtractor[\_X] | 浮点等不需要做立即数拓展和选通的ExuUnit前连接的模块。 | 源操作数数量为  ImmExtractor 1个  ImmExtractor\_4 3个  ImmExtractor\_8 2个  内部没有立即数拓展和源操作数选通模块，数据直通out = in |

注：selImm编码表中的立即数拓展目标都是32位，而ImmExtractor拓展目标是64位。在ImmExtractor中使用立即数拓展时，将selImm编码表中拓展的32位结果符号拓展到64位。

* + 1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| XLEN | 香山处理器支持数据运算最大位宽 | 恒等于64 |
| VADDRLEN | 香山处理器使用的虚拟地址位宽 | 恒等于39 |
| numSrc | RS 模块中 uop 需要的源操作数个数 | 根据 RS 后连接的不同FU需要的源操作数不同，numSrc 可以为1，2，3 |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| io\_uop\_ctrl\_srcType\_X | I | RS | ImmExt | 2 | uop 的第 X 个源操作数的类型，当类型为立即数时会使用 selImm 对应的拓展方式做立即数扩展。  X的取值范围为[0, numSrc) |
| io\_uop\_ctrl\_selImm | I | RS | ImmExt | 4 | uop 的类型，用于指定源操作数选通和立即数拓展方式。 |
| io\_uop\_ctrl\_imm | I | RS | ImmExt | 20 | 来自 uop 对应的原始指令中或是被融合指令中的立即数。这是立即数拓展前的数据。 |
| io\_uop\_psrc\_{0, 1} | I | RS | ImmExt | 8 | 复用信号。lui-load指令融合时，用来保存32位立即数基址(高20位来自lui，低12位来自load指令)的高12位，低20位在imm信号中。  仅有LoadImmExtractor模块使用。 |
| io\_data\_in\_X | I | RS | ImmExt | XLEN | 来自uop对应源操作数寄存器的数据。 |
| io\_data\_out\_X | O | ImmExt | RS | XLEN | 用于ExuUnit运算的数据。这是立即数拓展和源操作数信号选通的结果。 |
| jump\_pc | I | RS | ImmExt | VADDRLEN | 跳转指令所在的PC值（虚地址） |
| jalr\_target | I | RS | ImmExt | VADDRLEN | 立即数跳转指令的目标PC值（虚地址） |

* + 1. 补充说明
  1. AgeDetector
     1. 功能
        1. 功能概述

AgeDetector 模块维护 RS 中存有的 uop 的新旧顺序，输出当前最旧的 uop 对应OH向量。

* + - 1. 功能描述

AgeDetector 模块用状态矩阵寄存器 age 保存 uop 的新旧顺序，状态矩阵的行列数都与 RS 中的entry数量相同。

isEnqueued(i)表示 RS 中第 i 个位置的uop入队，isDeqed(i)表示 RS 中第 i 个位置的uop出队。enqueuePortBefore(i, j) 表示第 i 和 第 j 个 entry 正在同时入队。且第 i 个 entry对应的入队端口在第 j 个 entry 的端口前面。对坐标(i, j)对应的状态位，AgeDetector 模块根据以下规则计算新的状态矩阵age\_next。

* 当 RS 中第 k 个 uop 出队时，
  + age\_next(k, j)=0
    - 第 k 个 entry 不比第 j 个旧，
    - 特殊地，j = k时，age\_next(k, k)=0表明第 k 个 entry 无效，符合预期。
  + age\_next(i, k)=age(i, i) || (isEnqueued(i) && !isDeqed(i))
    - 第 i 个 entry 是否比第 k 个 entry 旧，取决于第 i 个 entry 是否存在，或第 i 个 entry 是否正在入队，
    - age(i, i)表明第 i 个 entry 已有指令，
    - isEnqueued(i)表明当前第 i 个 entry有指令正在入队，
    - !isDeqed(i)表明当前第 i 个 entry没有指令正在出队，
    - 特殊地，i = k时，按age\_next(k, k) = 0处理，表明第 k 个 entry 无效。
* 否则，当 RS 中第 k 个 uop 入队时，
  + age\_next(k, j)=!age(j, j) && !enqueuePortBefore(j, k)
    - 第 k 个 entry 是否比第 j 个 entry 旧，取决于第 j 个 entry 是否不存在，且是否不在第 k 个 entry 入队端口前的端口入队。
    - 特殊地，j = k时，由于age(k, k)=0，enqueuePortBefore(k, k)=0，age\_next(k, k)=1，表明第 k 个 entry 将有效，符合预期。
* 否则，RS 第 k 个 uop 没有变化，
  + age\_next(k, j) = age(k, j)
    - 需要注意的是，实际上age矩阵只使用一半。对age\_i\_j，
    - 当i <= j时，age\_i\_j寄存器实际存在
    - 否则，age\_i\_j将映射为age\_j\_i寄存器的相反值ImmExtractor 有多个特化版本，具体功能如下所示。
    1. 整体框图
    2. 接口列表
       1. 接口定义使用的参数列表

|  |  |  |
| --- | --- | --- |
| **参数** | **含义** | **取值** |
| numEntries | RS模块配置的entry数量 | 随不同RS模块的配置而变化 |
| numEnq | RS模块配置的单周期可入队的uop数量 | 随不同RS模块的配置而变化 |

* + - 1. 接口列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信号** | **方向** | **源** | **目的** | **位宽** | **描述** |
| clock | I | 1 |  | AgeDetector |  |
| reset | I | 1 |  | AgeDetector |  |
| io\_enq\_X | I | numEntries | RS | AgeDetector | RS 入队 entry 的 OH 向量。  X的取值范围为[0, numEnq) |
| io\_deq | I | numEntries | RS | AgeDetector | RS 出队 entry 对应多个 OH 向量的或 |
| io\_out | O | numEntries | AgeDetector | RS | 最旧的 entry 对应的 OH 向量 |

* + 1. 补充说明

无