中国科学院计算技术研究所 INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES
中国开放指令生态（RISC-V）联盟 China RISC-V Alliance
北京开源芯片研究院 BEIJING INSTITUTE OF OPEN SOURCE CHIP

香山（昆明湖架构）向量扩展的设计和实现

张紫飞[1] 胡轩[1] **张梓悦**[1] 唐浩晋[1] 何逸飞[2] 肖飞豹[2] 付丹阳[3]
张娄峄[4] 曹泽文[5] 贾志杰[1] 刘泽昊[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]大连理工大学 [4]北京大学 [5]中国科学院微电子研究所

2023 年 8 月 24 日

目录

- RISCV向量拓展
- 香山向量实现方案
- 向量拓展的优化
- 向量验证
- 总结

目录

RISCV 向量（Vector）扩展

- 独立的向量寄存器
- 向量 CSR
- 向量指令
  - 设置向量CSR + 访存 + 计算
- 向量循环：
  - 设置向量 CSR
  - 向量访存 Load + 向量计算 + 向量访存 Store
  - 更新指针
  - 判断循环

v0
v1
...
v31

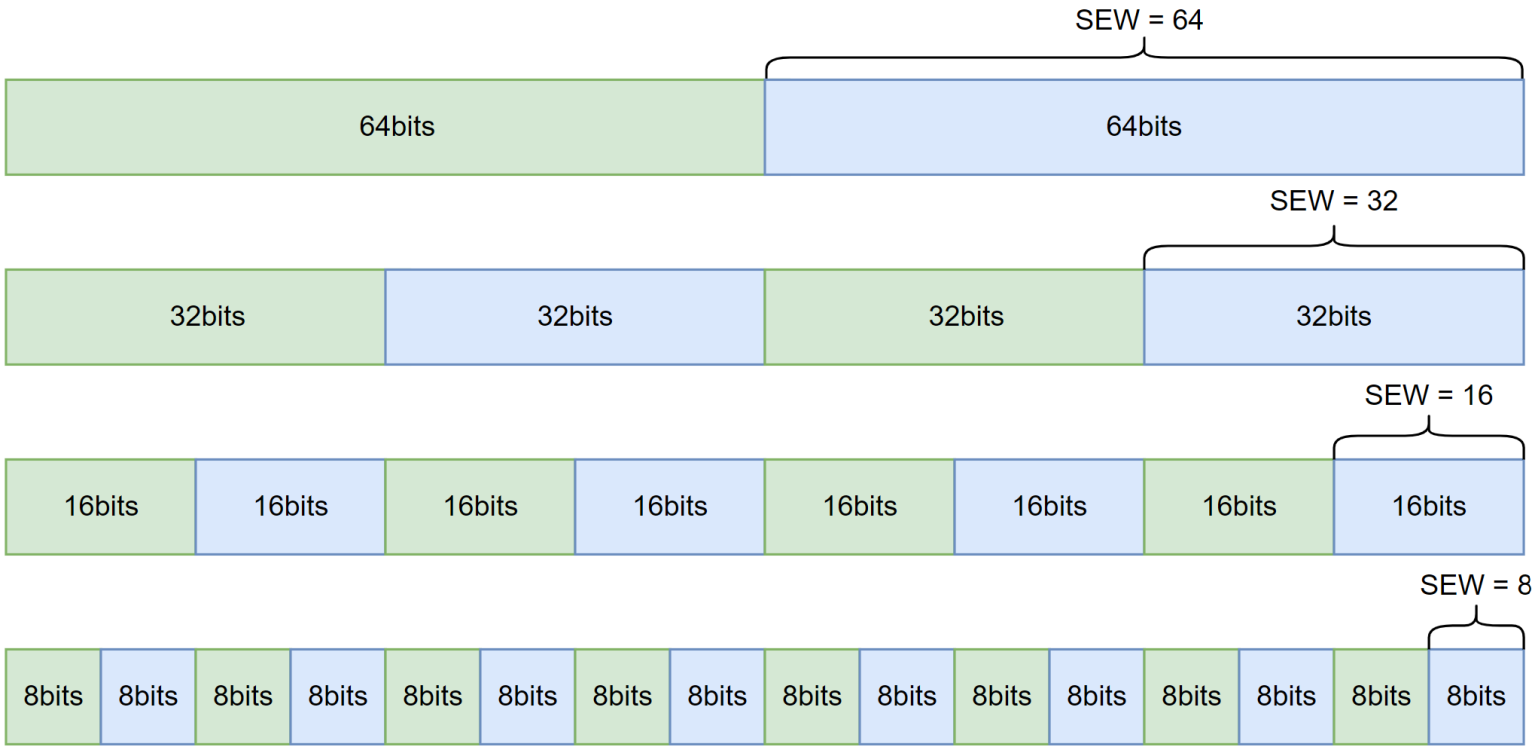
向量寄存器堆v0-v31

vl
vtype
vlenb
vstart
vxsat
vxrm
vcsr

新增向量 CSR

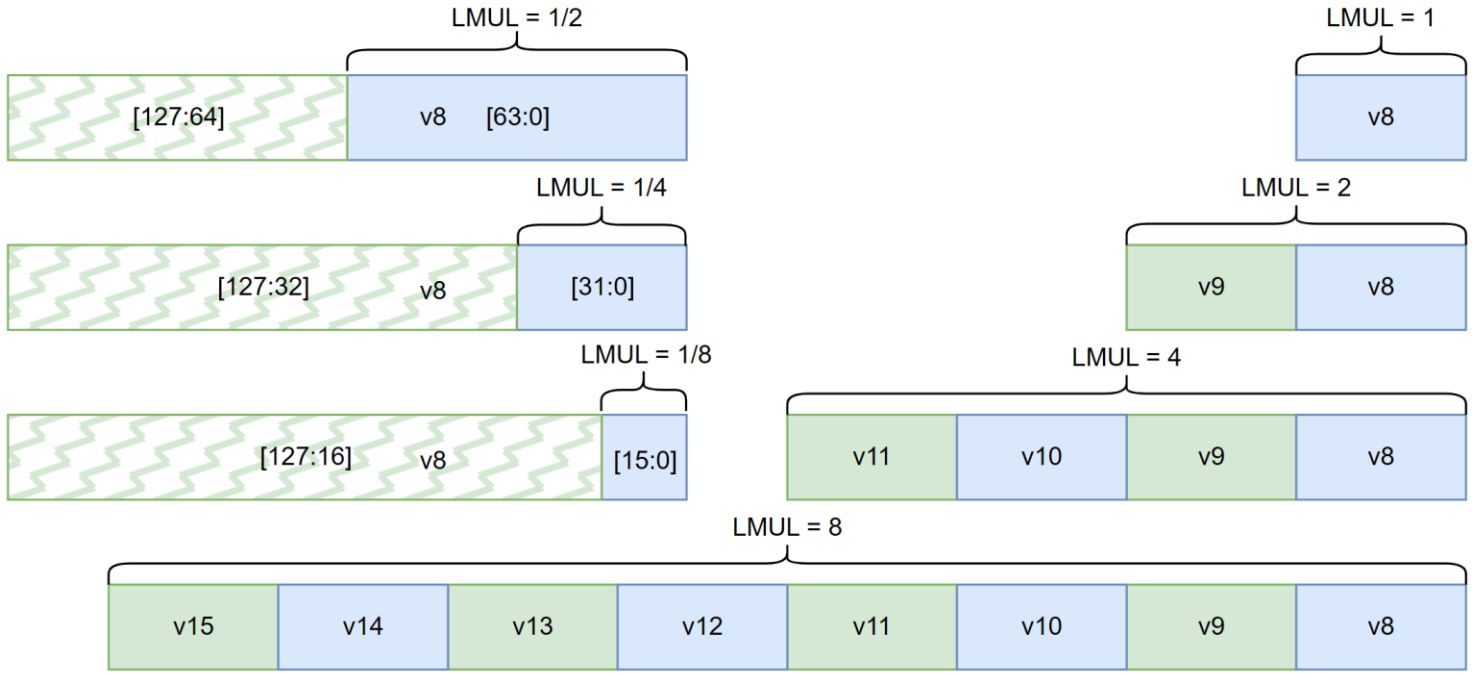
向量计算操作

- SEW(Selected Element Width)
  - 指定向量元素运算的位宽
    - 可取值为8、16、32、64
  - 由配置指令中的vtype域设置



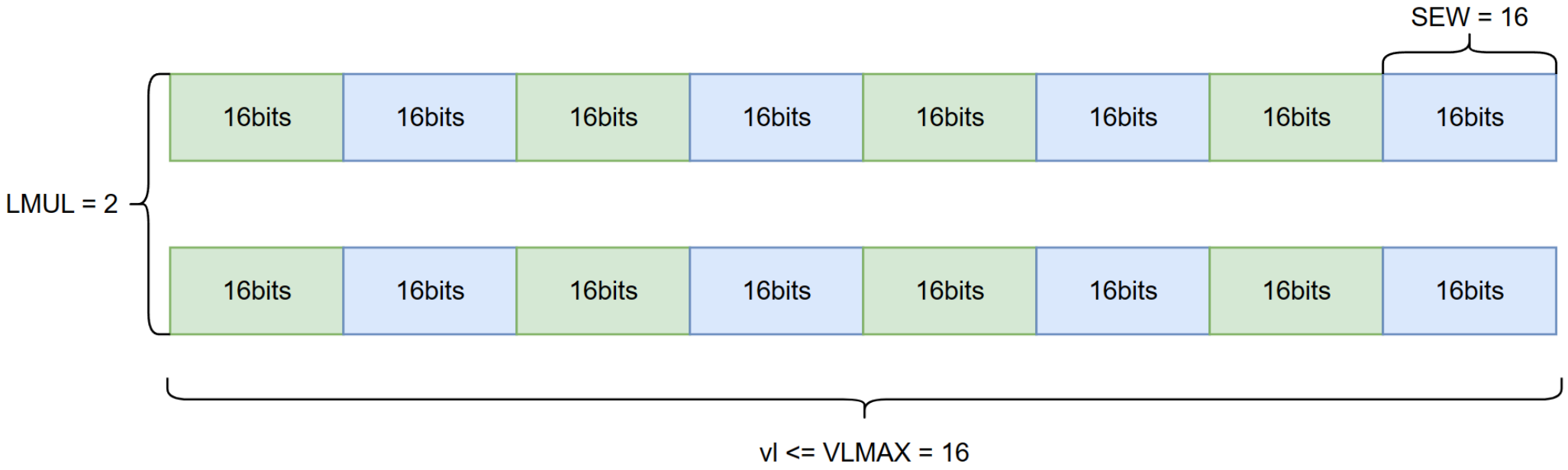
向量计算操作

- LMUL (Vector Register Group Multiplier)
  - Case1：将连续多个寄存器合并为寄存器组
    - 可取值为1、2、4、8
  - Case2：拆分单个向量寄存器
    - 可取值为1/2、1/4、1/8
  - 由配置指令中的vtype域设置



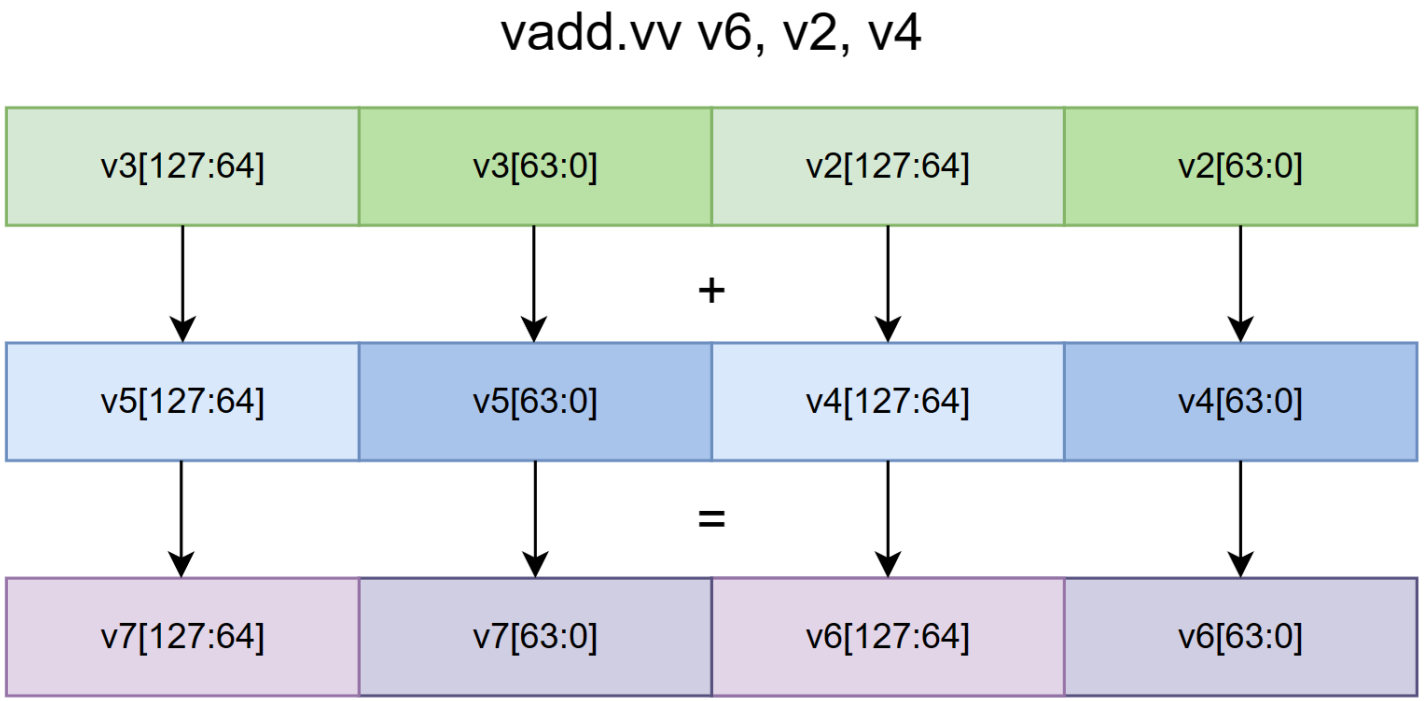
向量计算操作

- VL (Vector Length)
  - 进行向量运算的元素数目，小于等于VLMAX (VLEN*LMUL/SEW)
  - 由配置指令中的vl域设置



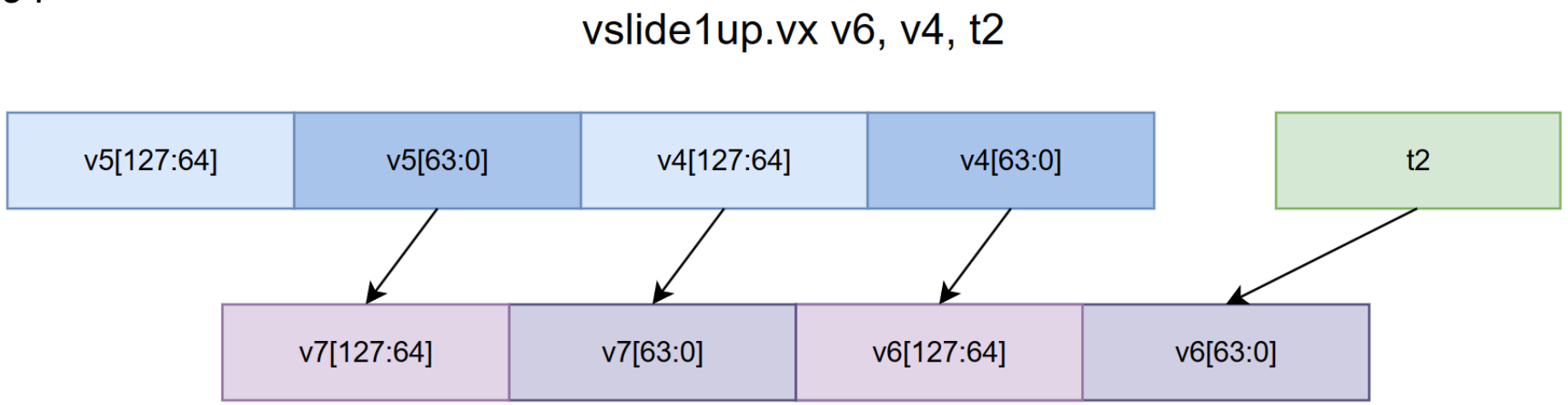
向量计算样例

- VADD指令
  - 两组向量寄存器进行加法运算，将结果写入目标向量寄存器组
  - LMUL = 2
  - SEW = 64
  - vl = 4



向量计算样例

- VSLIDE1UP指令
  - 向量寄存器间、标量向量寄存器间移动数据至对应项
  - LMUL = 2
  - SEW = 64
  - vl = 4



向量扩展特性

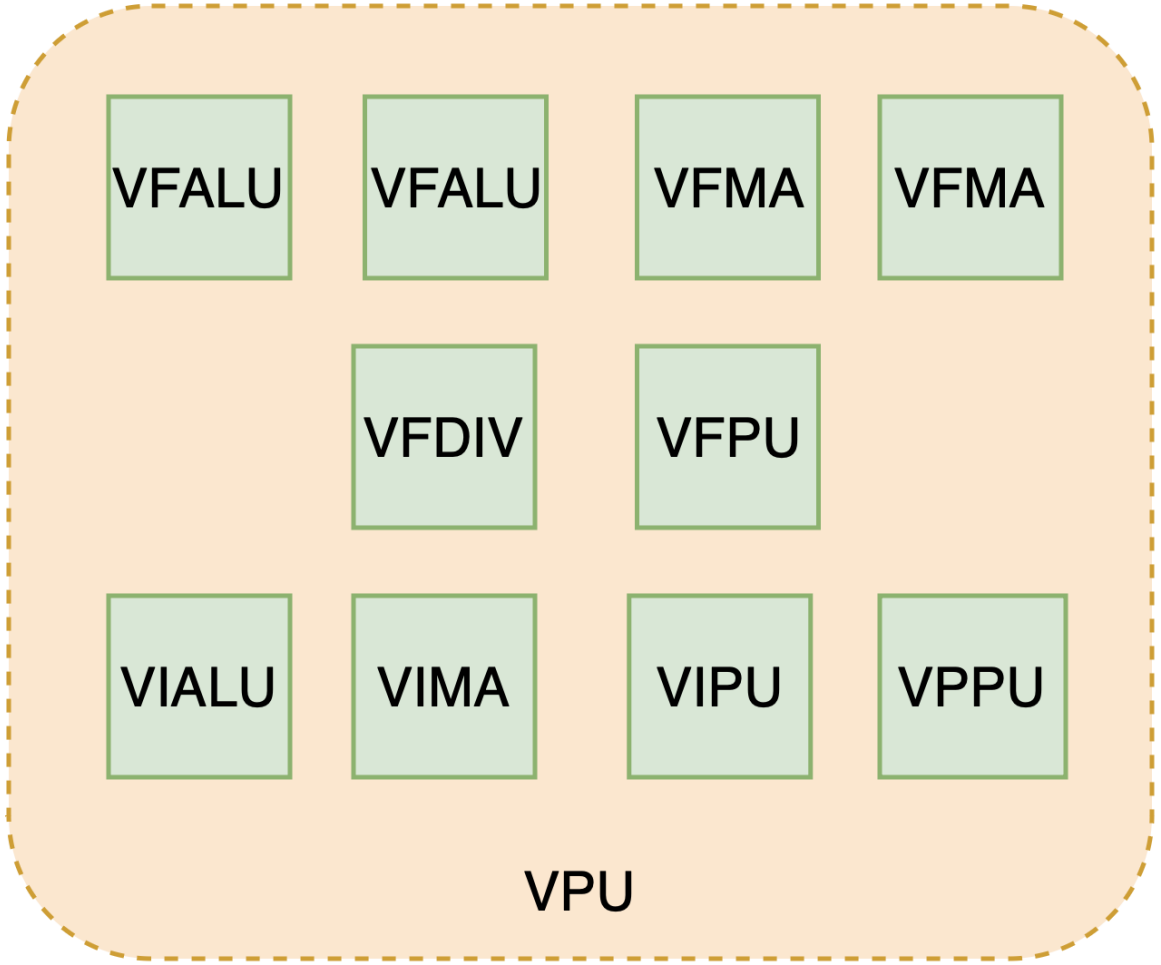
- 动态配置数据位宽、数量、舍入模式
- 一条指令操控多个向量寄存器
- 使用 v0 作为谓词寄存器实现谓词操作
- 种类丰富的计算类型和访存模式

目录

香山向量设计架构

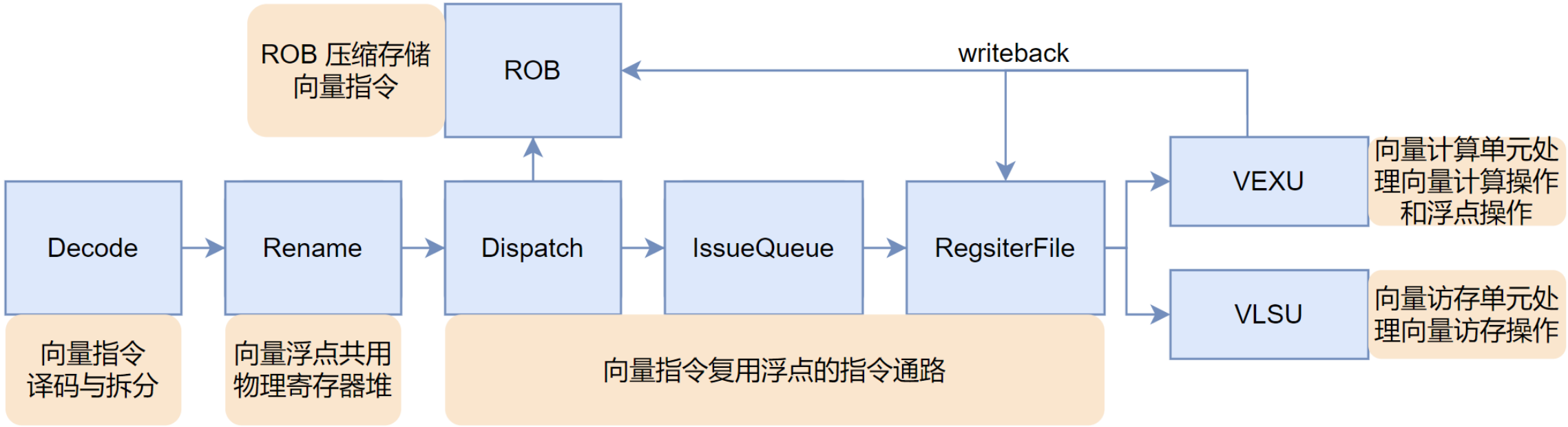
- 昆明湖向量扩展规格
  - 兼容 RISCV Vector 1.0 向量指令集扩展
  - VLEN： 128
  - 向量计算单元开源仓库 (YunSuan)

https://github.com/OpenXiangShan/YunSuan



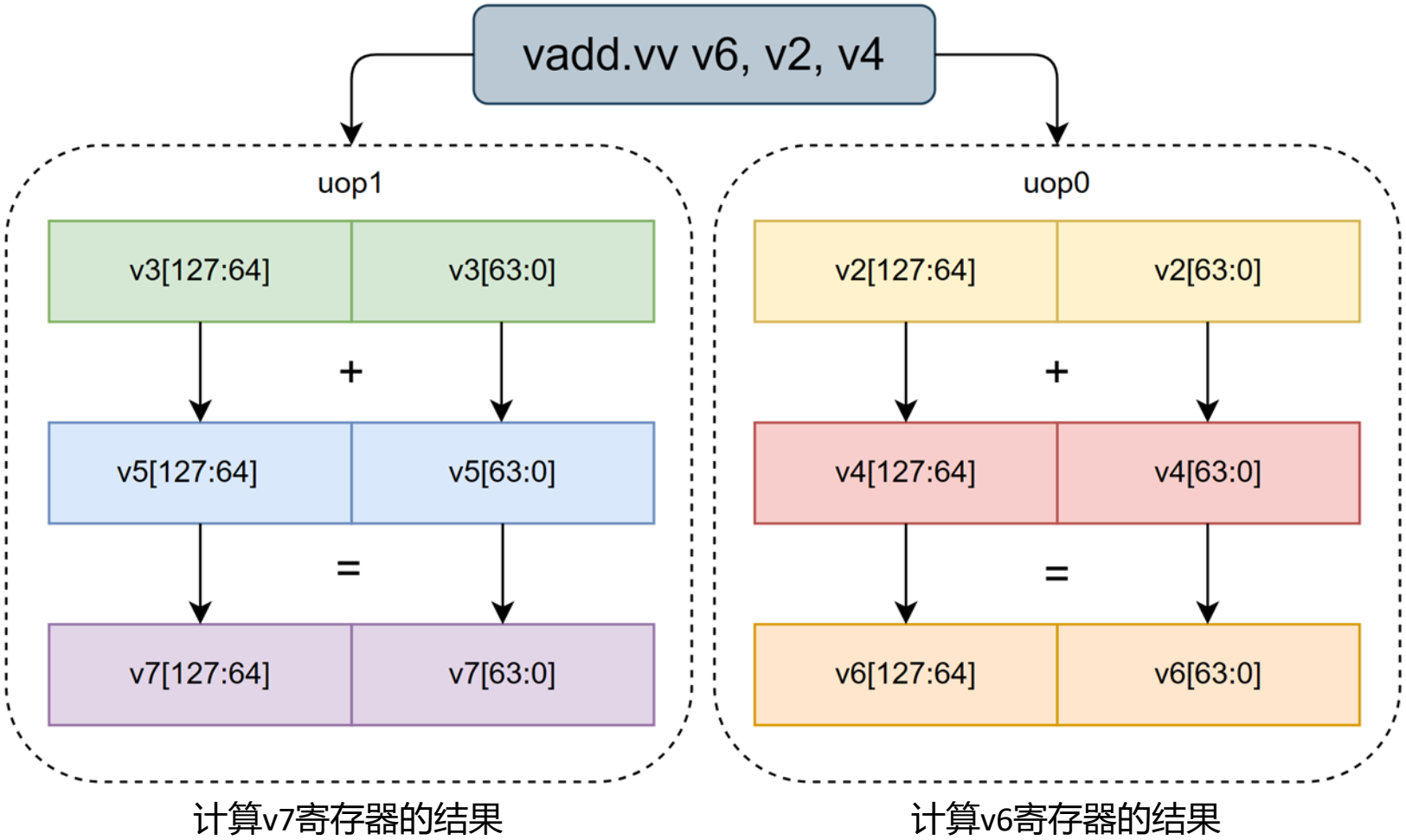
香山向量设计架构

- 紧耦合式向量扩展设计
- 复用流水线：译码/分派/乱序/执行



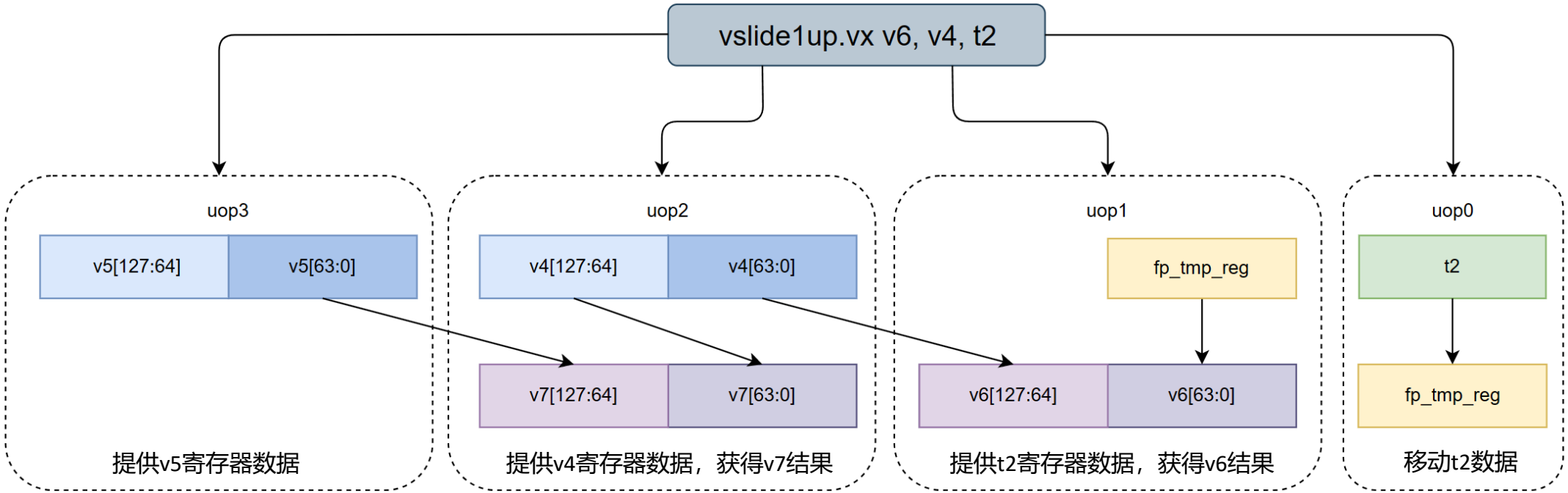
向量简单指令的拆分

- VADD指令
  - 以向量寄存器为粒度拆分指令
  - 拆分出来的向量微操作称为uop
  - 并行计算uop中的元素



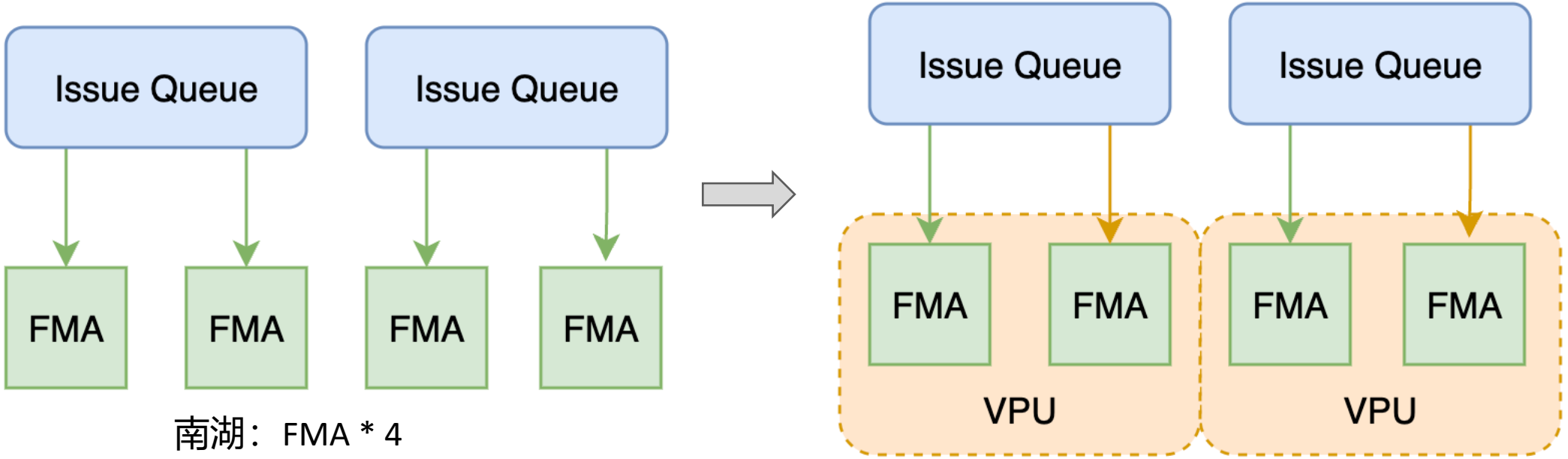
向量复杂指令拆分

- VSLIDE1UP指令
  - 定义一个临时浮点寄存器
  - 添加一条标量寄存器到临时浮点寄存器的move指令



向量浮点共用发射队列与计算单元

- 南湖浮点单元：FMA * 4
- 昆明湖向量单元：VPU * 2 cover FMA * 4
- 维持原有标量浮点的同时，实现向量计算加速

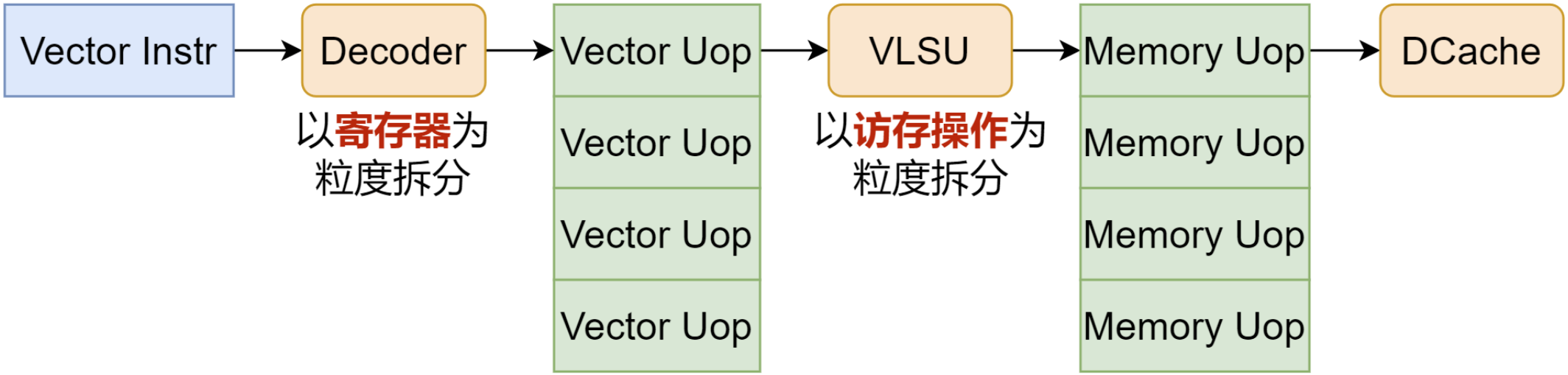


南湖：FMA * 4

昆明湖：VPU * 2

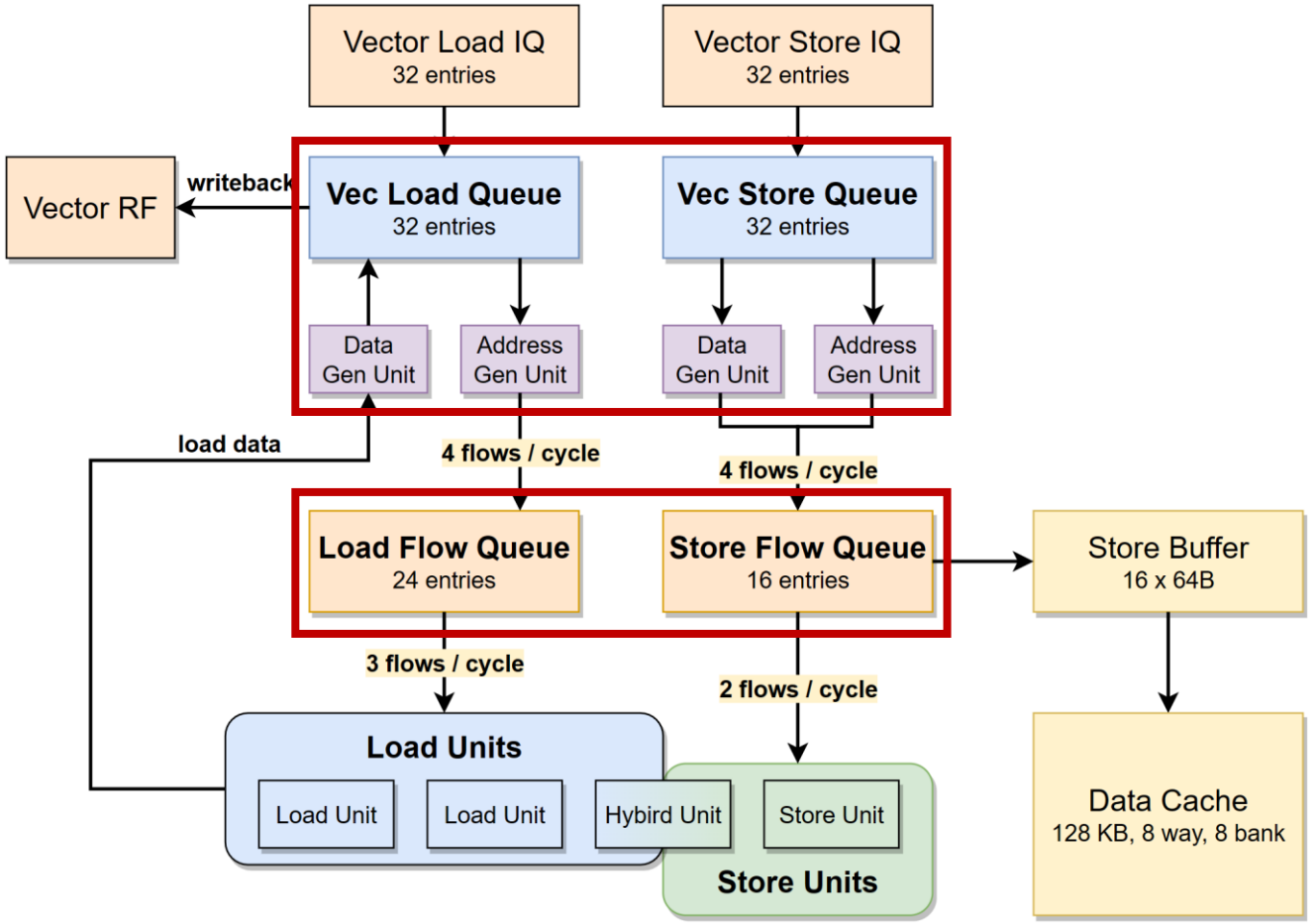
向量访存操作

- 向量访存指令的两阶段指令拆分
  - 译码单元以 **寄存器** 为粒度进行拆分
  - 向量访存单元以 **访存操作** 为粒度进行再拆分

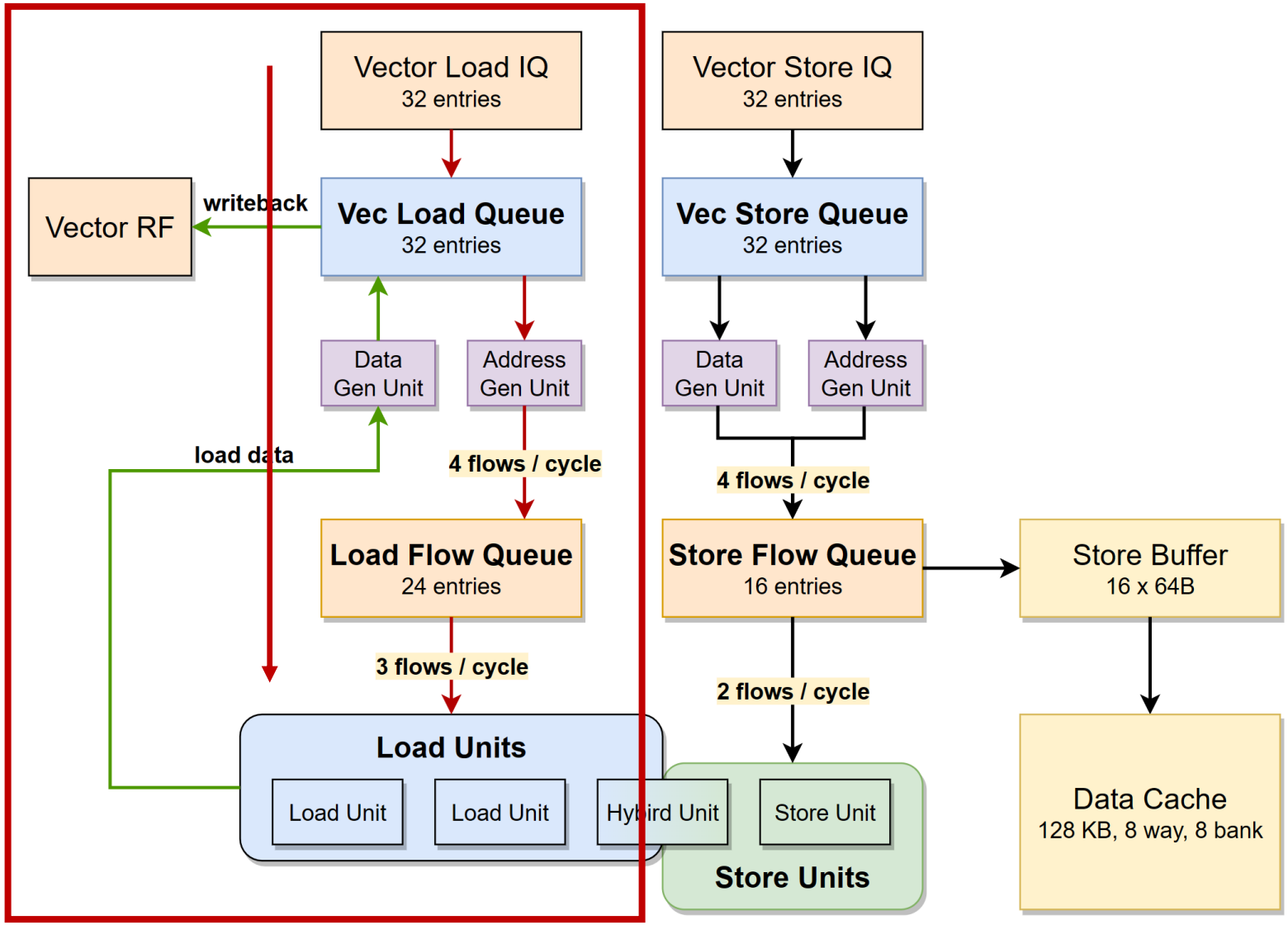


香山处理器向量访存实现概述

- 流水线**紧耦合**的向量访存
  - 复用标量 Load / Store Unit
  - 与标量共用 Data Cache
- 支持所有向量访存指令
- 主要功能部件
  - **Vector load / store queue**
  - **Load / store flow queue**
- **正在持续开发**

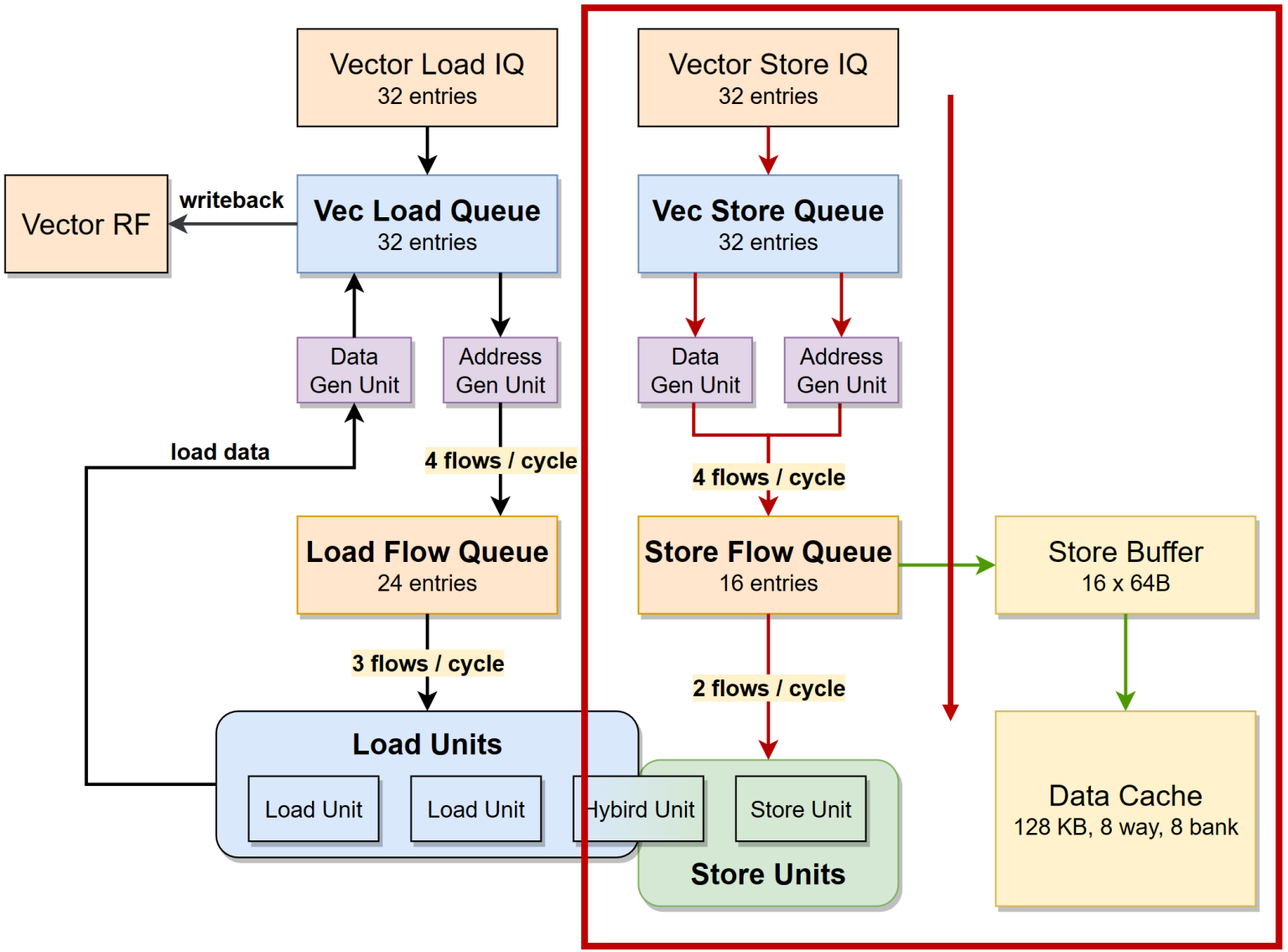


向量访存通路 — Load



- Vector Load **Issue Queue**
  - 按**目的寄存器粒度**拆分 uop
- Vector **Load Queue**
  - 维护**访存顺序**等信息
  - 以**访存粒度**拆分出flows
  - **拼接** Load Unit 返回的数据
  - 以**寄存器为粒度**写回
- Load **Flow Queue**
  - 保存**单次访存粒度**的 flow

向量访存通路 — Store



- Vector Store **Issue Queue**
  - 按**数据寄存器粒度**拆分 uop
- Vector **Store Queue**
  - 维护**访存顺序**等信息
  - 以访存粒度**拆分 store data**
- Store **Flow Queue**
  - 保存 **store data**
- Store **Write Data**
  - 等待 address 和 data **就绪后**写入数据至 sbuffer

向量例外分类

- 按阶段分类
  - 译码时判断
  - 执行时判断
- 按指令类型
  - 配置指令
  - 计算指令
  - 访存指令

处理阶段	例外分类	具体内容
译码阶段	指令编码字段非法	vm、simm等字段不合法
	vtype信息非法	vill字段不合法
		操作数eew不合法
		操作数emul不合法
	向量寄存器非法	向量寄存器组编号未对齐
		目的寄存器与v0 mask寄存器重叠
		目的寄存器与源寄存器重叠
计算阶段	vstart非法	vstart非零
		实现中计算指令不允许vstart非零执行
访存阶段	vstart非法	vstart超过当前配置的vlmax
		访存指令允许vstart非零执行
	访存地址权限非法	每个访存操作都需要进行权限检查
	访存地址非对齐	每个访存操作都需要进行地址对齐检查

向量例外处理

- 译码时例外：
  - **与标量相同**，进入ROB，不进入乱序流水
- 向量计算例外：
  - **等到所有微操作写回**，报例外，刷新流水
- 向量访存例外：
  - **等待所有微操作写回**，**维护 vstart 值**等，报例外，刷新流水

目录

重命名加速向量配置指令

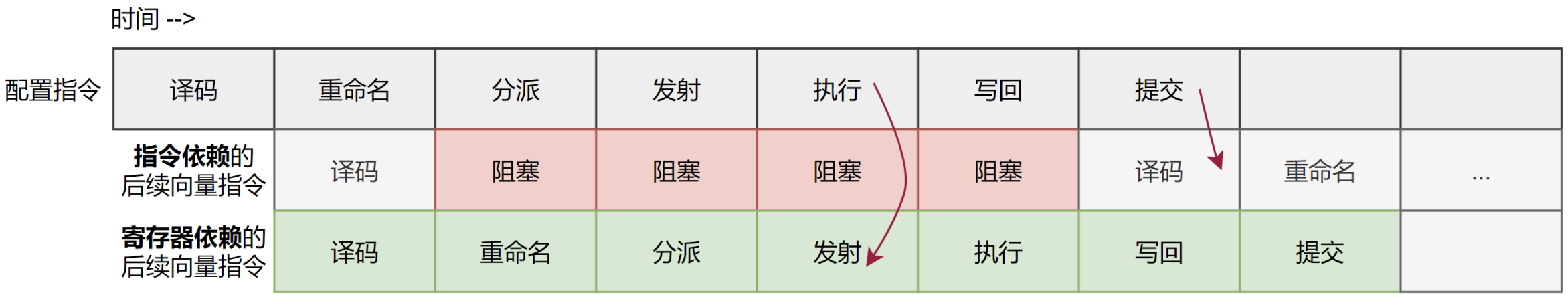
- 向量配置指令修改 **向量长度寄存器(vl)** 与 **向量类型寄存器(vtype)**
- 后续向量指令 **依赖于** 向量配置指令 / VCSR

- 漫长的阻塞时间

新增逻辑寄存器建立数据依赖

- 添加 **32号 浮点寄存器**： vl + vtype
- **依赖 指令/VCSR -> 依赖 32号浮点寄存器**

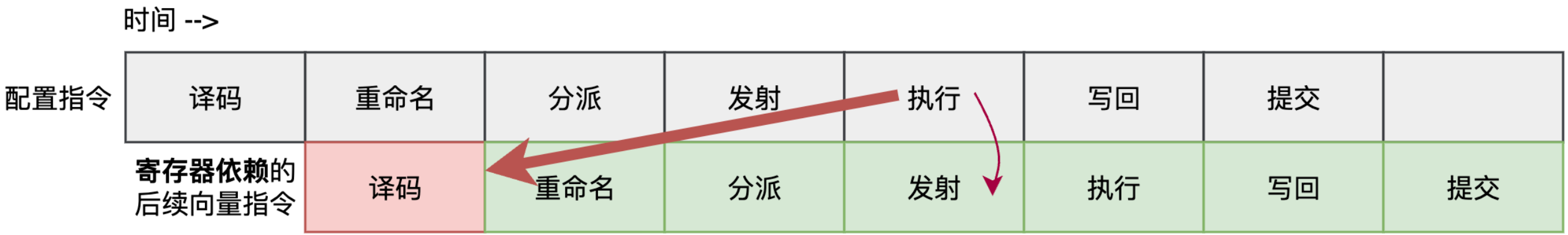
- 缩短等待阶段



向量配置操作-未解决的问题

- 向量指令拆分数量依赖于 vtype.vlmul

- **译码阶段需要配置指令的 vtype 结果**



推测性更新 vtype

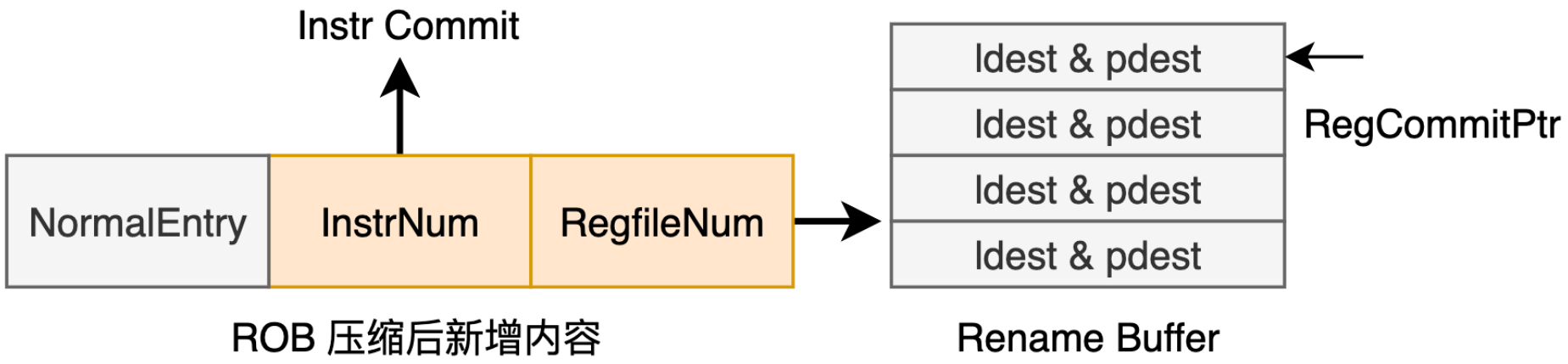
- 常用配置指令 **vtype 来自于立即数**

- **推测性type**：译码阶段维护推测状态的 vtype
- **更新**：配置指令译码阶段更新 vtype
- **错误恢复**：建立 vtype 表，应对错误恢复

向量指令拆分的数量爆炸

- 问题1：一条向量指令拆分多个向量微操作（1-64）
- 问题2：向量微操作内容相似

- **ROB 压缩**：一个 ROB 项存储多个指令
  - 一个ROB项存储多个向量微操作
- **Rename Buffer**：解耦寄存器提交和指令提交
  - 解决单指令多寄存器提交的问题



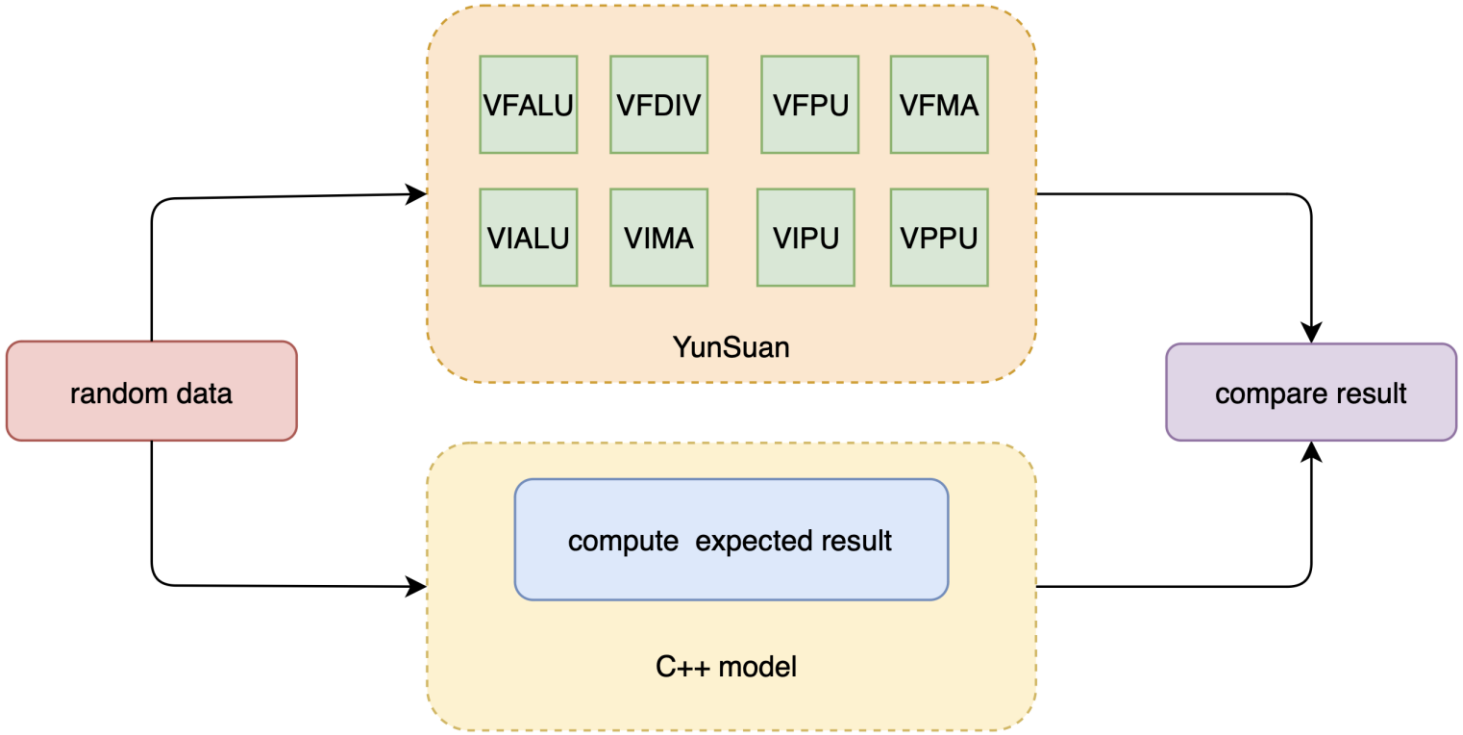
目录

向量验证

- 单元验证：
  - 针对 YunSuan 的随机输入单元验证框架
- 系统验证：
  - 基于 riscv-dv 的随机指令生成器
  - 针对每个指令的程序测试集
  - 针对每个例外的程序测试集
- 计划中：
  - OpenCV 等
  - 自动向量化：SPEC CPU 2006等
  - 带有向量扩展的程序检查点（Checkpoint）

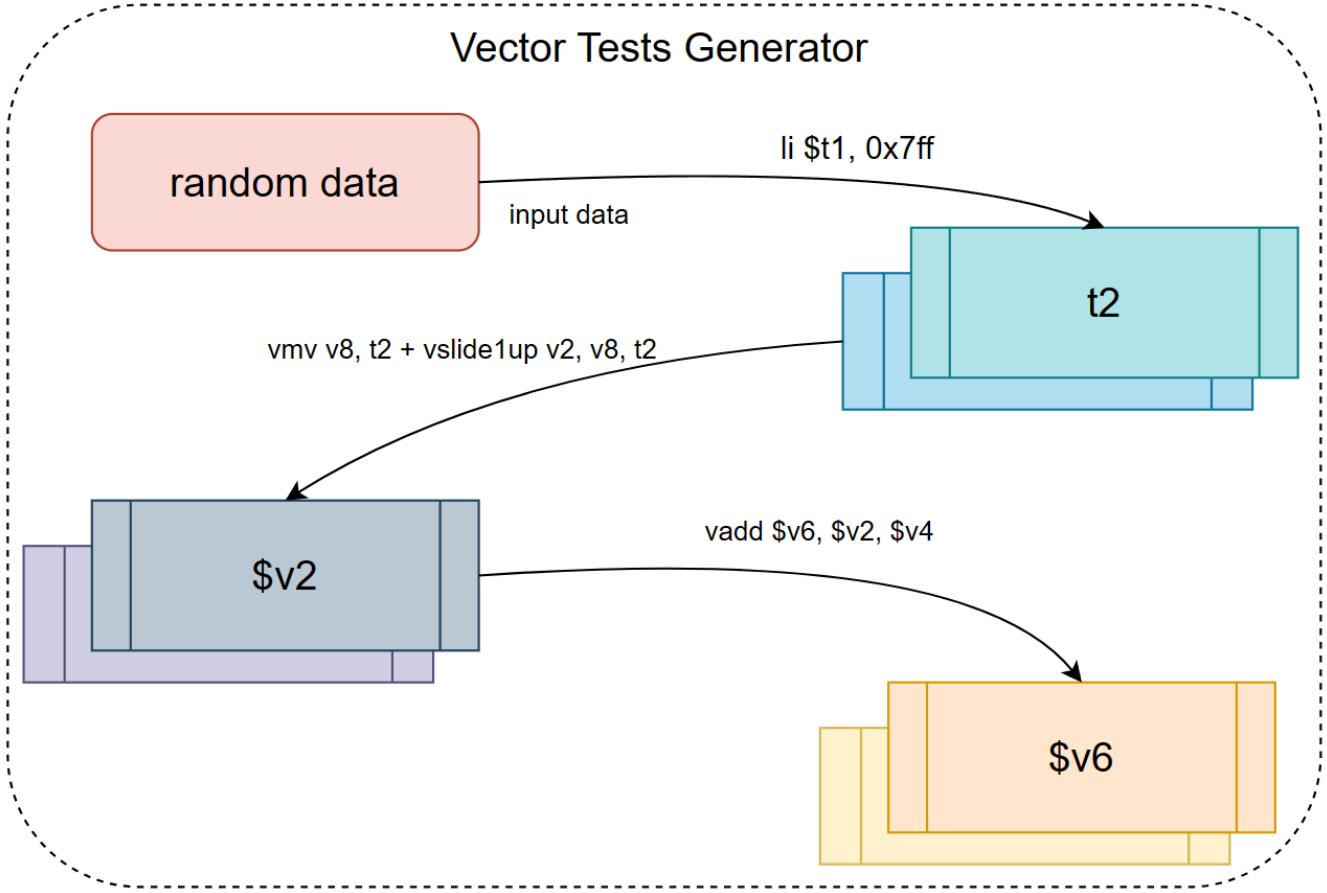
向量单元验证

- 采用在线对比差分测试框架
  - 生成激励
    - softfloat-testfloat
    - random
  - 传入向量运算单元进行计算
  - 传入C++模块计算期望结果
  - 比较计算结果



向量系统验证

- NEMU实现向量拓展
  - 支持RISCV Vector 1.0 向量指令集扩展
- 基于riscv-dv实现随机指令生成器
  - 生成每一条指令对应的随机测试
- 使用DiffTest框架与NEMU进行比对
  - 每一个周期比对向量寄存器、向量CSR



目录

- RISCV向量拓展
- 香山向量实现方案
- 向量拓展的优化
- 向量验证
- 总结

总结

- RISC-V 向量扩展介绍
- 香山向量扩展架构设计
  - 配置、计算、访存、例外
- 向量扩展的优化
  - 配置依赖、ROB 压缩、解耦寄存器与指令提交
- 向量验证
  - 随机验证、单元验证、系统验证

敬请批评指正！