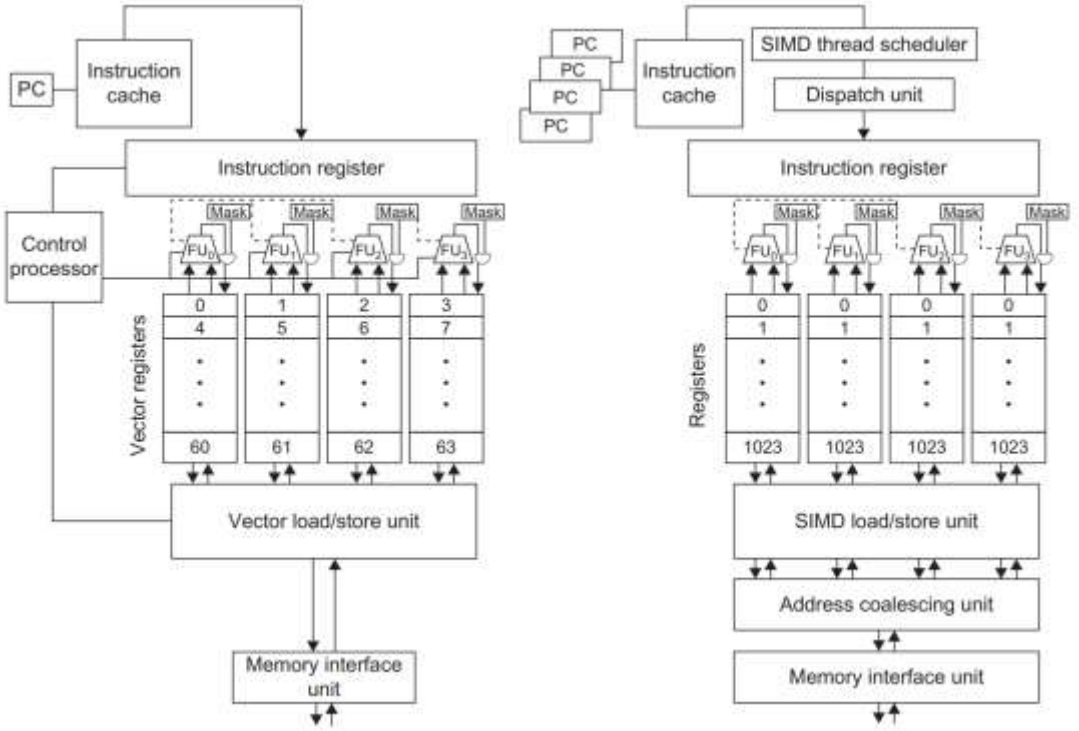
“乘影”：开源通用GPU指令集架构介绍

清华大学集成电路学院

于芳菲

向量处理器与GPGPU

- 典型向量处理器与典型SIMT架构在计算、访存、分支、任务控制、寄存器堆较为相似
- 动机：将向量lane与GPGPU thread一一对应



	计算单元	访存单元	分支控制	任务控制	寄存器堆	整体结构
Vector	Vector Lane	Gather/Scatter	Mask Registers	Control Processor	Vector Registers	Vector Processor
GPGPU	SIMD Lane	Global load/store	Predicate Registers	Thread Block Scheduler	SIMD Lane Registers	Multithreaded SIMD Processor

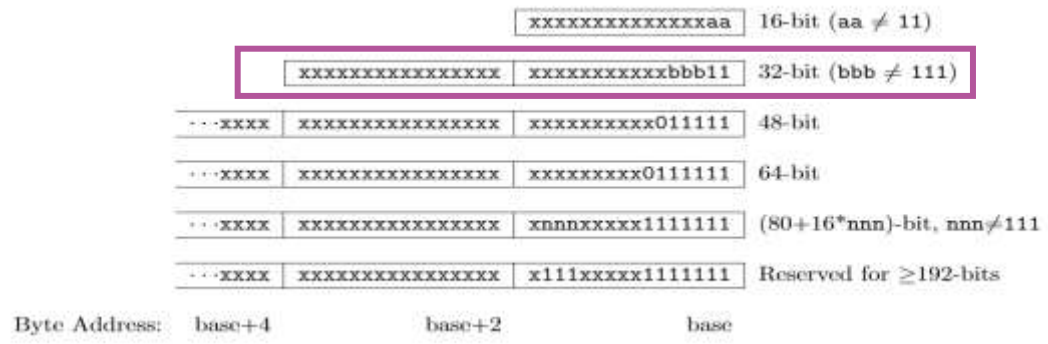
*图片来自《计算机体系结构 量化研究方法》

“乘影”GPGPU指令集 - 设计概览

- 指令集：RISC-V + 向量扩展
  - GPGPU与向量处理器的结构有许多相似之处
  - 大部分GPGPU所需指令可以由RISC-V向量扩展集提供
- GPGPU额外的特性和功能需求：
  - 线程束的分支和同步、多线程束执行
  - 按OpenCL模型要求，线程在运行时能够获取自身索引号
  - 更高的指令表达能力，以提升执行效率

扩展	指令类型	乘影
V(zve32f)	Configuration-Setting	部分支持
	Loads and Stores	支持
	Integer Arithmetic	支持
	Floating-Point	支持fp32
	Fix-Point	参照需求添加
	Reduction、Mask、Permutation	参照需求添加
I		支持
M		支持
A		支持
zicsr		支持
F(zfinx)		支持

“乘影”GPGPU指令集-扩展方案



- 指令编码空间有限?
  - 官方给出的扩展空间仅包括4个opcode7
- 乘影GPGPU采用RISC-V 32-bit指令字作为基础
  - 舍弃RISC-V 16-bit指令，释放低2位作为GPGPU的指令编码空间
  - RISC-V 32-bit指令bbb空间111之外的部分也可以作为扩展指令空间
- GPGPU 64-bit指令分两种实现方式:
  - 作为RISC-V 32-bit指令的前缀，实现对RISC-V现有指令的兼容
  - 利用16-bit释放出来的00,10,01实现完全自定义的32-bit和64-bit GPGPU指令
- 以上方法既可以兼容RISC-V 32-bit指令字，同时可以高效的利用指令字空间构建GPGPU指令集

“乘影”GPGPU指令集 – 标量指令

- 覆盖范围：RV32IMA_zicsr_zfinx RV64
- I：基本指令
  - 整数基本运算、访存、线程束整体分支、函数调用
  - *未支持ecall和ebreak指令
- M：乘法指令
- A：原子指令
  - 约束多线程对同一地址的读写操作，描述的是单个线程行为。
- zicsr：CSR寄存器操作
  - 线程和线程块索引在启动线程束前存入特定CSR以供读取
- zfinx：单精度浮点指令
  - 不设标量浮点寄存器，标量浮点数同样存入整数寄存器
- RV64：更改了部分指令语义，支持64位操作

“乘影”GPGPU指令集 – 向量指令

- 在V扩展的zve32f子集上进行裁剪与修改，使之符合SIMT编程模型：
- 支持基本运算和访存指令，保证线程标量操作均存在向量版本指令
- GPGPU计算方式特性考量：
  - 未支持元素数目和宽度的变长特性，固定元素数目32+宽度32位
  - 未支持向量元素间数据交换（gather, scatter, shuffle等）
  - 写回数据位宽小于32时（如掩码计算、读字节），仍采用32位对齐方式写回

“乘影” GPGPU指令集 – 寄存器堆

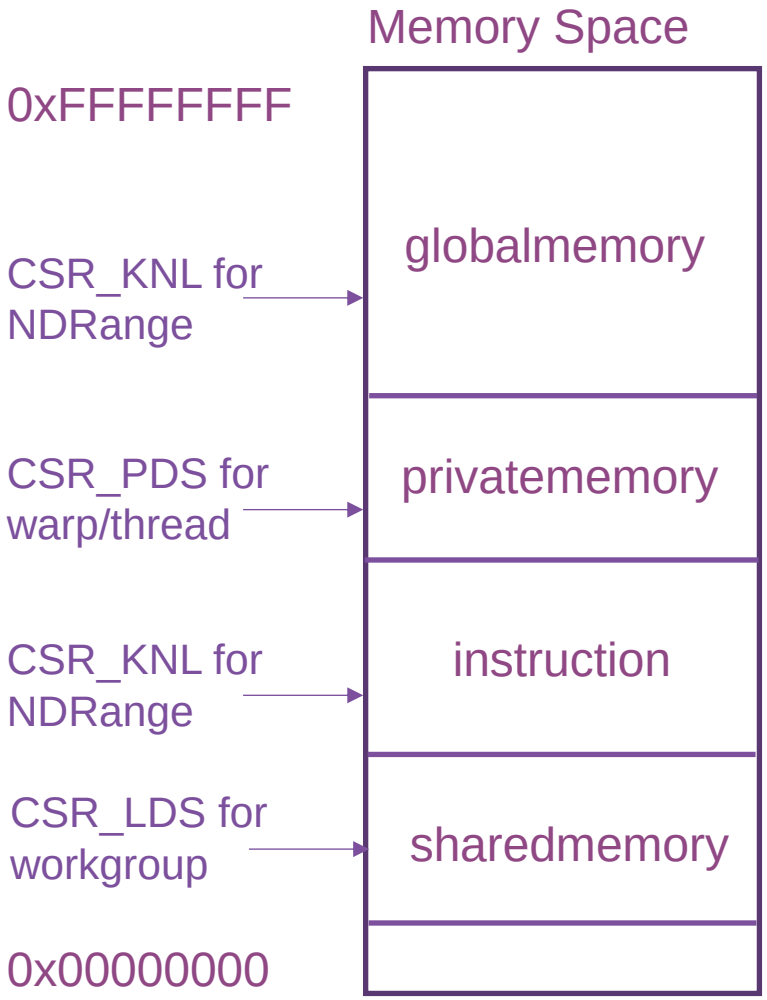
- 每个warp有64个32bit sGPR，256个1024bit vGPR(num_thread=32时)。标量寄存器由同一warp内的线程共享。

- 特殊寄存器：x0为0寄存器，x1为返回pc寄存器，x2为栈指针寄存器、x3为shared memory基址，x4为private memory基址
- CSR：添加warp id、workgroup id、kernel metadata baseaddr等

- ABI：约定8个sGPR和32个vGPR为函数参数，16个vGPR为返回值

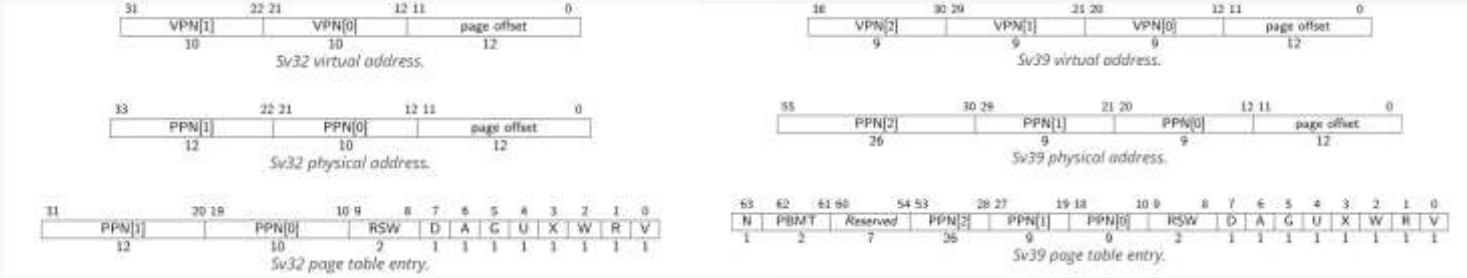
“乘影” GPGPU指令集 – 存储模型

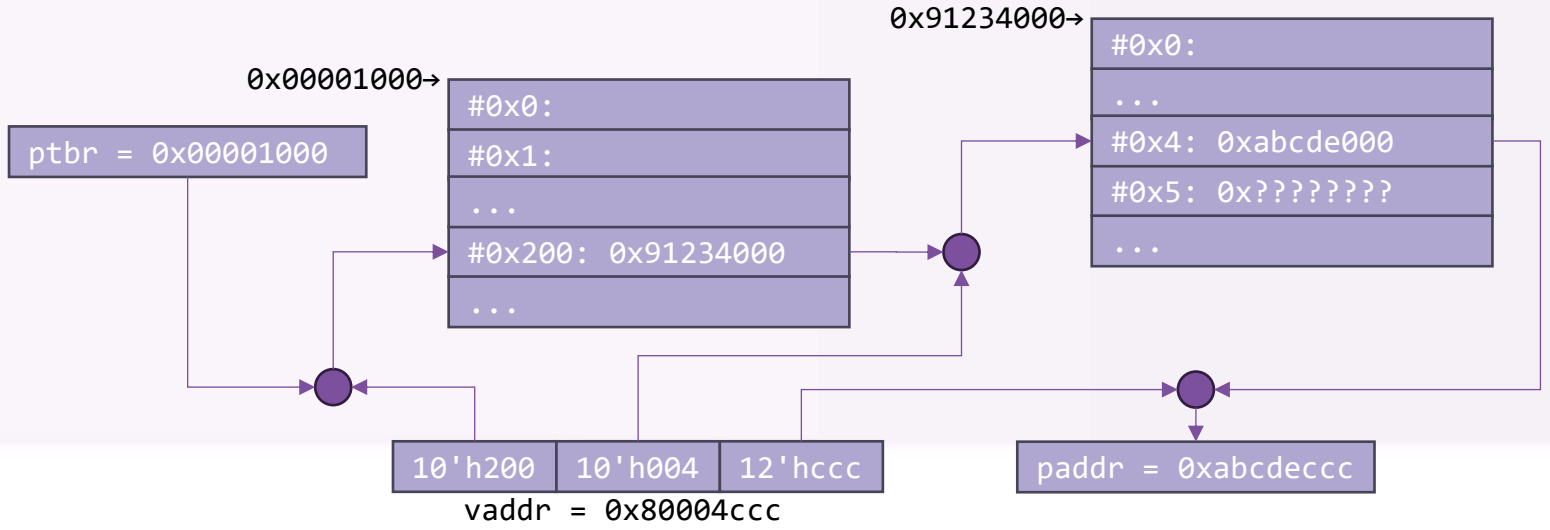
- 不同线程层次对应不同存储层次：
  - warp地址空间：PrivateMemory（栈）
  - workgroup地址空间： SharedMemory(LDS)
  - kernel地址空间：GlobalMemory（数据）、ConstantMemory（数据、指令）
- 驱动和线程块调度器完成地址分配
- 针对不同空间数据采用不同指令/地址范围进行访问
  - 分配的内存基址初始化到CSR里
  - 访存指令可以根据指令中所指示的存储位置或根据地址范围访问对应地址空间。



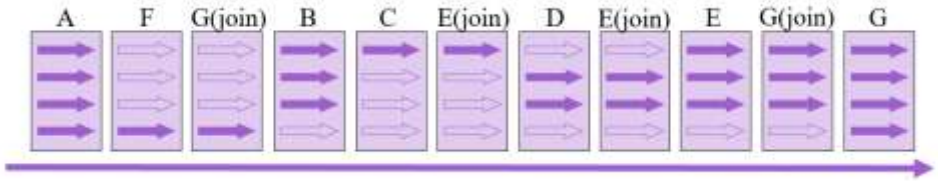
虚拟内存：Sv32/Sv39内存规范

- RISC-V定义了多种不同长度的虚拟地址与相应页表体系
  - Sv32：虚拟地址32位 / 物理地址34位，两级页表(10+10)
  - Sv39：虚拟地址39位 / 物理地址56位，三级页表(9+9+9)

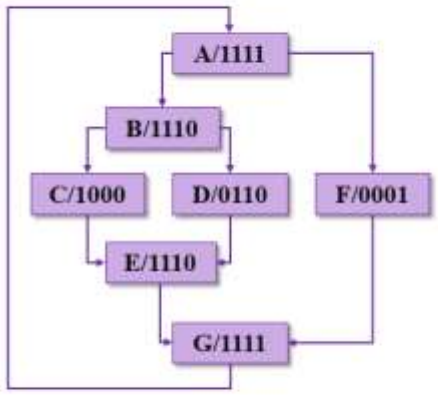




自定义指令：线程束分支与汇合



- 使用自定义指令和硬件分支同步栈
- 重汇聚pc设置指令：setrpc
  - 将即将出现的分支的重汇聚PC写入rpc CSR寄存器
- 线程束分支指令： vbeq、vbge等
  - 从CSR中取出上一次写入的rpc值
  - 当发现各个线程判断结果不一致、出现分歧（THEN和ELSE）时，将取出的rpc和当前掩码压栈、将线程数多的分支的PC值和掩码压栈，按线程数少的分支掩码执行线程数少的分支 （线程数量相同时先执行ELSE分支）
- 线程束汇合指令：join
  - 分支执行到此处，若栈顶rpc与当前执行PC相同时，弹出栈顶信息，将PC置为PC，掩码置为栈顶掩码；若栈顶rpc与当前执行PC不同，则不做任何操作
- 在未分歧情形下，简化压栈和汇合操作，只执行一个分支



(a) 程序样例

rPC	PC	掩码(mask)
-	-	-

(b) 栈初始状态

rPC	PC	掩码(mask)
G	G	1111
G	B	1110

(c) 分支1发生后

rPC	PC	掩码(mask)
G	G	1111

(d) F段程序执行完毕后

rPC	PC	掩码(mask)
G	G	1111
E	E	1110
E	D	0110

(e) 分支2发生后

rPC	PC	掩码(mask)
G	G	1111
E	E	1110

(f) C段程序执行完毕

rPC	PC	掩码(mask)
G	G	1111

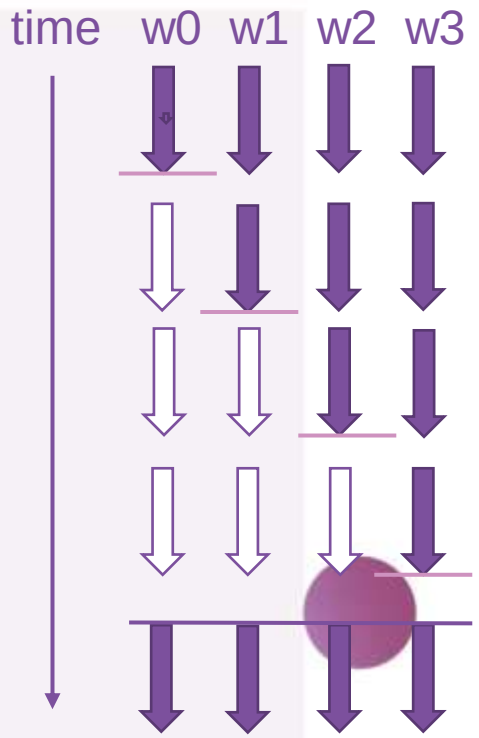
(g) 分支2执行完毕

rPC	PC	掩码(mask)
-	-	-

(h) 分支1执行完毕

自定义指令：线程同步和退出

- SIMT编程模型需求线程间的数据交换
  - 显式插入栅栏（barrier）
  - 线程到达栅栏后需等待线程块内所有线程均到达，方可继续执行



- 栅栏指令：barrier、barriersub
  - 可阻拦特定内存或线程范围的访存操作
  - 可指定解除栅栏所需的到达线程数目

- 线程退出指令：endprg
  - warp的核函数退出，回收PC控制器、指令缓冲、寄存器堆、共享内存等资源

自定义指令：额外访存指令

- RISC-V向量扩展的访存指令设计与GPGPU需求并不一致
- RISC-V向量扩展的长立即数版本向量访存指令缺失
  - 添加vls12系列指令，支持向量寄存器基址+12位立即数偏移量访存
- OpenCL需求访问多类地址空间，RISC-V向量扩展并无相关支持
  - 增加地址空间标签访问的功能，由硬件映射为真实地址

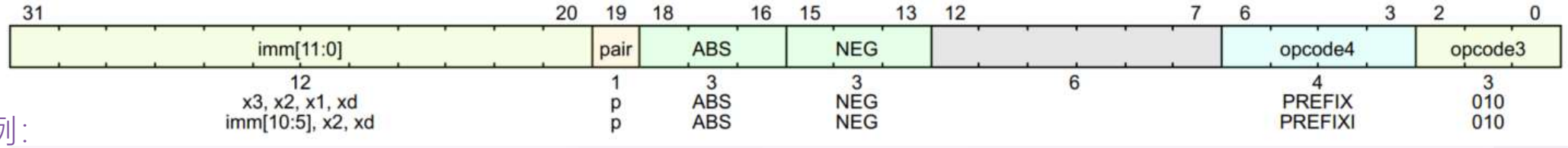
地址空间	地址计算
Private	addr=(vs1+imm)*num_thread_in_wg+thread_idx+csr_pds 或者 addr=([vs1:vs1+1]+imm)*num_thread_in_wg+thread_idx+csr_pds
Local	addr=(vs1+imm)+csr_lds 或者 addr=([vs1:vs1+1]+imm)+csr_lds
Global	addr=(vs1+imm)+csr_gds 或者 addr=([vs1:vs1+1]+imm)+csr_gds
default	根据访存指令定义进行地址计算

自定义指令：前缀指令

- 问题1
  - RISC-V原始寄存器编码仅有5位，32个可用寄存器
  - OpenCL内建向量数据类型在“乘影”架构中会占用多个向量寄存器，容易发生寄存器溢出
  - 向量指令操作数较多，立即数仅有5位，只能表示-16~15的整数
- 问题2
  - 32位寄存器的架构只能访问4GB的内存空间
  - 将寄存器全部扩展为64位带来额外的硬件开销和资源浪费
  - 混合32位64位数据类型在编译器支持上存在问题
- 解决方案
  - 设计前缀指令补充语义
  - 支持寄存器对操作
  - 按需支持超过32个寄存器的寻址及64位操作，保留不扩展的兼容性

自定义指令：计算前缀指令

- 用于扩展其后指令的寄存器编码及立即数编码
- 兼具寄存器对操作、计算结果取反、取绝对值语义
- 编码扩展方式：
  - Imm12字段携带下一条指令的寄存器\立即数高位编码
  - 寄存器扩展(PREFIX)：四个操作数寄存器扩展高3位
  - 立即数扩展(PREFIXI)：两个操作数寄存器扩展高3位、立即数扩展高6位
  - 支持8位寄存器编码（256寄存器）、11位立即数
- 寄存器对拼接：
  - 采用偶对齐，相邻两个32位寄存器拼为64位



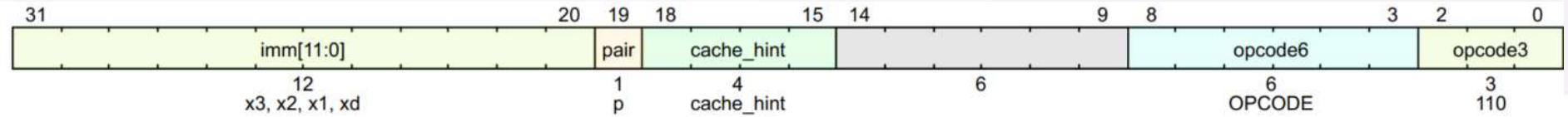
示例：

```
prefix 0, b000_001_000_010, 0, 0
vadd.vx v16, v20, x8
```

等价于vadd.vx v80, v20, x40

自定义指令：访存前缀指令

- 设计访存前缀指令，指示后续一条指令的访存空间、缓存策略、读取数据宽度等信息。
- 包含寄存器扩展和寄存器对信息
- 访存空间：
  - private memory, global memory, local memory
- 数据宽度：
  - 32bit(1word), 64bit, 96bit, 128bit。
  - 若宽度大于32，则从连续的地址空间读取数据存入相邻寄存器或从相邻寄存器读出数据存入连续地址空间
- 预留cache hint字段



自定义指令：异步拷贝指令

- 配合张量计算单元，提升数据搬移能力
- cp_dma指令
  - 小规模异步拷贝指令，用于4-16字节
- cp_dma_bulk指令
  - 大规模异步拷贝指令，大于等于16字节，从标量寄存器中读取数据规模
- cp_dma_tensor指令
  - 从全局内存拷贝一个张量类型的数据到共享内存，从标量寄存器中读取地址、维度等信息
- cp_dma_barrier指令
  - 线程束执行到这条指令且异步搬运完成后，执行其后的指令

自定义指令：计算指令

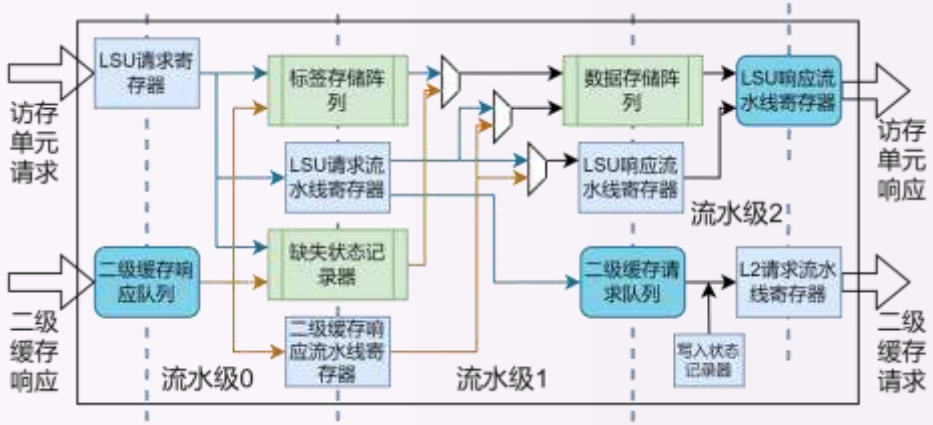
- 添加vadd12指令，以12位立即数为源操作数
  - 降低寻址、立即数装载/加减、跳转时的立即数扩展频率

- 添加矩阵乘加MMA指令和特殊函数（指数）指令
  - 支持不同规模、不同精度的矩阵乘加运算
    - D = A × B + C
  - 数据类型：
    - FP32、FP16、BF16、FP8（E4M3）、FP8（E5M2）
    - INT8、INT4、Binary
  - 排列方式：行优先或列优先

description				
A/B type	shape	C/D type	A.layout	B.layout
FP32	m8n8k8	FP32	A矩阵在A.layout =0时行优先，A.layout =1时列优先	B矩阵在B.layout =1时行优先，B.layout =0时列优先
	m16n8k8			
	m16n8k16			
FP16	m8n8k8	FP16		
	m16n8k8			
	m16n8k16			
BF16	m8n8k8	FP32		
	m16n8k8			
	m16n8k16			
	m8n8k8	FP32		
	m16n8k8			
	m16n8k16			
FP8(E4M3)	m16n8k32	INT32		
FP8(E5M2)	m16n8k32	INT32		
INT8	m8n8k16	INT32		
	m16n16k16			
INT4	m8n8k32	INT32		
	m16n8k32			
	m16n8k64			
Binary	m8n8k128	INT32		

“乘影” GPGPU指令集 – 缓存一致性

- 在RISC-V弱存储模型的基础上，“乘影”进一步部署了释放连贯性指导的缓存一致性（RCC）。让SM私有缓存间具备一致性功能的同时，避免了硬件一致性协议带来的高昂硬件成本和运行时带宽开销。原子指令也处理成对单个线程的行为。



微架构操作	④全局无效化	③全局无效化	④全局无效化
RVWMO 连贯性语义	.aq标识符	.rl标识符	FENCE
附带的RCC 一致性操作	“获取”和“释放”	“释放”	“获取”和“释放”

“乘影”GPGPU与其它ISA对比

ISA	AMD RDNA	AMD GCN	NVIDIA PTX	Intel GEM	Imagination PowerVR	Vortex RISC-V	乘影 RVV
内存模型	GDS, LDS Constants Global	GDS, LDS Constants Global	Shared, Texture Constants Global	由软件管理	Global Common Store Unified Store	Shared Global	Private, LDS Constants Global
线程模型	Workgroup Wavefront 32/64线程	Compute Unit Wavefront 64线程	Grid/CTA Warp	Root Thread Child Thread	USC 32线程	Compute Unit Wavefront	Workgroup Warp(Wavefront) 32线程
指令字长	32/64位	32/64位	128位 (SASS, Ampere)	128位		32位	32/64位
寄存器堆	向量和标量 256 VGPRs 106 SGPRs	向量和标量 256 VGPRs 102 SGPRs	标量 (虚拟指令集)	向量 128 GRFs	128-bit 向量	标量	向量和标量 256 VGPRs 64 SGPRs
同步	barrier wait_cnt	barrier wait_cnt	barrier membar	wait fence	fence	barrier flush	barrier (membar) fence

“乘影”GPGPU与其它ISA对比（续）

ISA	AMD RDNA	AMD GCN	NVIDIA PTX	Intel GEM	Imagination PowerVR	Vortex RISC-V	**乘影 RVV**
线程控制	endpgm指令 线程mask	endpgm指令 线程mask	谓词（predicate）	消息（Message）	谓词（predicate）	线程mask	**endprg指令 线程mask**
指令流和分支控制	branch 线程mask	branch 线程mask split/join	branch predicate	branch SPF Regs split/join	branch predicate	split/join	**branch vbranch/join**
ALU操作	算术 条件判断 位运算	算术 条件判断 位运算	算术 条件判断 位运算	算术 条件判断 位运算	算术 条件判断 位运算	算术 条件判断 位运算	**算术 条件判断 位运算**
访存操作	load store prefetch	load store prefetch	load store prefetch	load store	load store	load store	**load store**
其他	图形 Matrix Core 光线追踪	图形	图形 Tensor Core 光线追踪	图形 Matrix Engine 光线追踪	图形	仅实现Texture Sampling	**Tensor Core**

OpenGPGPU
乘影

THANK YOU

4th 第4届 2024
RISC-V 中国峰会
RISC-V Summit China