CMO指令介绍

——RISC-V Base Cache Management Operation ISA Extensions

PLCT实验室：史玉龙

Email：yulong@nj.iscas.ac.cn

引言
CMO是什么？
CMO做什么？
CMO实现进展？

引言——一点点的知识背景（别叫停）

- 内存和缓存

memory location是物理地址唯一标识的物理资源，agent是能够访问memory location的逻辑块，如RISC-V hart、I/O设备等，通常是load和store操作。

缓存是存在于agent和memory location之间的，可以减少平均内存延迟，可以代替memory location来满足agent的load和store操作的一种结构。

CMO是什么？

- 缓存管理操作是一组对内存层次结构中数据副本执行相关操作的指令。
- 可以直接对缓存的数据副本进行操作，也可以直接对memory location进行操作
- CMO指令按照操作可分为以下几类：

1、管理指令(management instruction)——其针对一组可以访问数据的代理，操作缓存数据副本

2、零指令(zero instruction)——将相关内存位置清零

3、预取指令(prefetch instruction)——告知硬件，给定的memory location中的数据将会被访问

- 本次介绍的是一组基本的CMO ISA指令扩展，操作对象是缓存块或与缓存块对应的memory location，称为缓存块操作(CBO)指令，分类如下：

1、Zicbom扩展定义了CBO.INVAL、CBO.CLEAN、CBO.FLUSH

2、Zicboz扩展定义了CBO.ZERO

3、Zicbop扩展定义了PREFETCH.R、PREFETCH.W、PREFETCH.I

CMO做什么？

- Zicbom扩展是定义了缓存块的管理指令，见下表：

RV32	RV64	Mnemonic	Instruction
√	√	cbo.clean base	Cache Block Clean
√	√	cbo.flush base	Cache Block Flush
√	√	cbo.inval base	Cache Block Invalidate

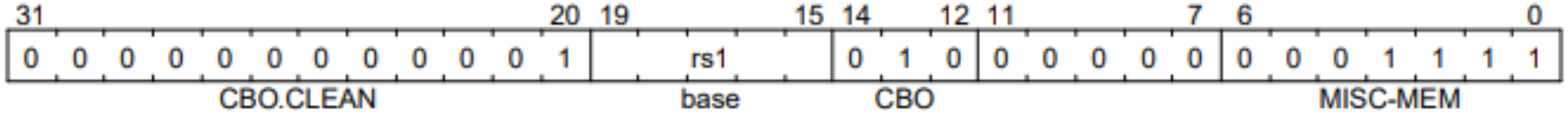
invalidate operation——释放缓存块的副本

clean operation——如果缓存块副本中的数据已被store操作修改，则该指令执行一个写入传输到另一个缓存或者内存

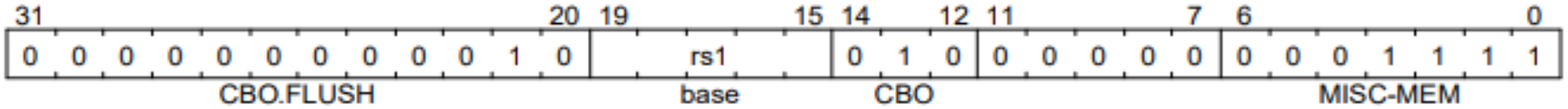
flush operation——执行invalidate operation之后的clean operation

Zicbom扩展

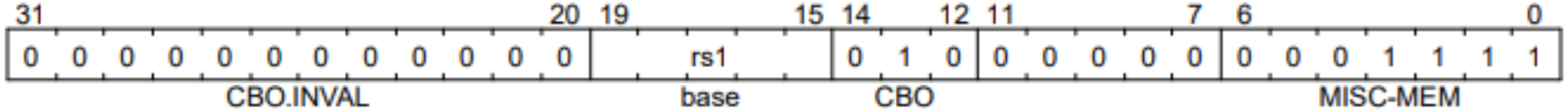
- cbo.clean指令格式——cbo.clean base



- cbo.flush指令格式——cbo.flush base



- cho.inval指令格式——cbo.inval base



- 这些指令对有效地址在rs1寄存器中指定的缓存块进行操作

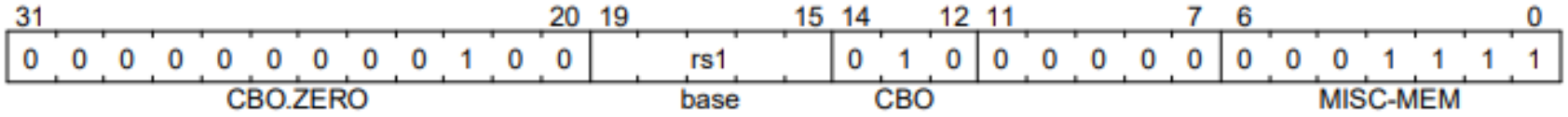
CMO做什么？

- Zicboz扩展定义了缓存块zero操作指令，见下表：

RV32	RV64	Mnemonic	Instruction
√	√	cbo.zero base	Cache Block Zero

zero operation——将0存储到与缓存块对应的空间中

该指令对缓存块或缓存块对应的内存位置操作，其有效地址在rs1中指定，指令格式为——cbo.zero



CMO做什么？

- Zicbop扩展是定义了缓存块预取指令，见下表：

RV32	RV64	Mnemonic	Instruction
√	√	Prefetch.i offset(base)	Cache Block Prefetch for Instruction Fetch
√	√	Prefetch.r offset(base)	Cache Block Flush
√	√	Prefetch.w offset(base)	Cache Block Invalidate

预取指令作为硬件的提示（告诉硬件软件即将执行特定类型的内存访问，三种类型分别为：instruction fetch、data read、data write。）

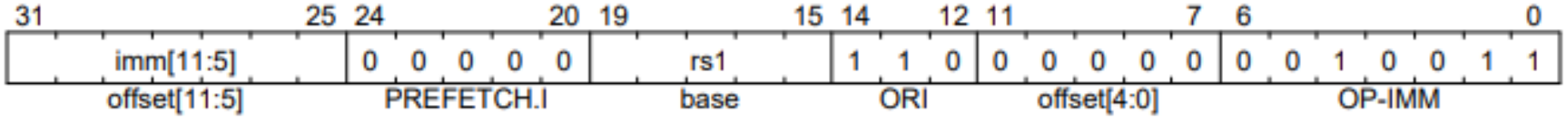
prefetch.i——告诉硬件，缓存块即将被instruction fetch访问

prefetch.r——告诉硬件，缓存块即将被data read访问，比如load

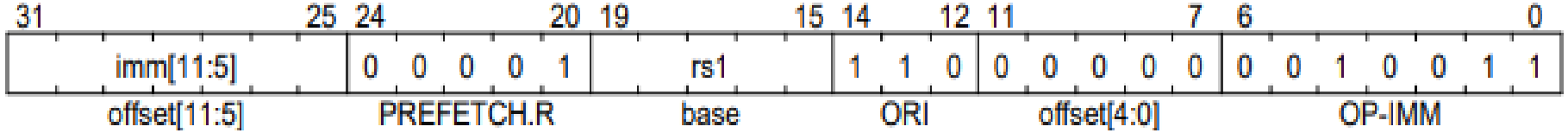
Prefetch.w——告诉硬件，缓存块即将被data write访问，比如store

Zicbop扩展

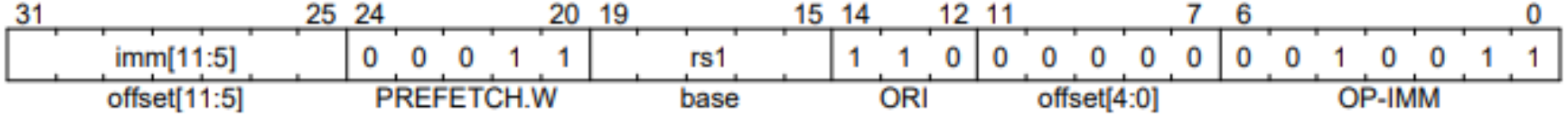
- prefetch.i指令格式——prefetch.i offset(base)



- prefetch.r指令格式——prefetch.r offset(base)



- prefetch.w指令格式——prefetch.w offset(base)



- 预取指令的有效地址是rs1中存放基地址和imm[11:0]之和

CMO实现进展——已实现

- 已实现binutils下汇编指令到二进制码映射关系：
- Patch0：https://sourceware.org/pipermail/binutils/2021-December/118909.html
- Patch1：https://sourceware.org/pipermail/binutils/2021-December/118906.html
- Patch2：https://sourceware.org/pipermail/binutils/2021-December/118908.html
- Patch3：https://sourceware.org/pipermail/binutils/2021-December/118907.html

以上patch添加了zicbom和zicboz扩展命令的基本实现和测试用例

- Patch4：https://sourceware.org/pipermail/binutils/2021-December/118910.html
- Patch5：https://sourceware.org/pipermail/binutils/2021-December/118911.html

以上patch添加了zicbop扩展命令的基本实现和测试用例

CMO实现进展——未实现

- CMO扩展依赖于环境配置的CSRs状态，目前CSR还未通过标注。
- 由3个CSRs控制CMO命令：menvcfg、senvcfg、henvcfg

bits	name	Description
[5:4]	CBIE	缓存模块invalidate指令使能 00：指令引发非法指令或虚拟指令异常 01：指令被执行，并且执行一个flush操作 10：reserved 11：指令被执行，并且执行一个invalidate操作
[6]	CBCFE	缓存模块clean和flush指令使能 0：指令引发非法指令或虚拟指令异常 1：指令被执行
[7]	CBZE	缓存模块zero指令使能 0：指令引发非法指令或虚拟指令异常 1：指令被执行

- gcc部分上游未见实际产出