SIPU RV Core常用指令归纳

1 RV Core简介

Vector registers：32个，VLEN定义了其最大bit数量，一般为256、512，或1024

Vector elements：ELEN定义了单个element的最大bit数量

单个Vector register可存放的element数量为：VLEN / ELEN

DLEN：512bit

EEW：effective element width，可用于source与destination element大小不等的情况

SEW：selected element width，一般情况下，EEW与SEW相等

LMUL：Vector length multiplier（1,2,4,8），Vector register group，可包含1个或多个Vector registers，作为单个指令操作数

EMUL：effective LMUL，由存放元素寄存器的个数定义

VLMAX：LMUL * VLEN / SEW，表示给定SEW与LMUL的情况下，单个Vector指令可以操作的操作数的最大元素个数

Mask register：占用1个Vector register

2 数据类型

2.1 Vector Data Types

RV Core中，Vector数据类型的命名规则为：

代码块

```
1   v DATA_TYPE ELEN LMUL _t
```

其中，DATA TYPE与ELEN常用的类型如下表所示：

int64	Signed long
uint64	Unsigned long
int32	Signed integer
uint32	Unsigned integer
int16	Signed short
uint16	Unsigned short
int8	Signed char
uint8	Unsigned char
float64	double
float32	float
float16	_Float16
bfloat16*	__bf16

LMUL常用数值如下表所示：

m1	LMUL = 1
m2	LMUL = 2
m4	LMUL = 4
m8	LMUL = 8
mf2	LMUL = 1/2
mf4	LMUL = 1/4
mf8	LMUL = 1/8

示例如下：

代码块

```
vint32m4_t，表示32bit有符号整形，且LMUL=4
```

2.2 Vector Mask Types

RV Core中，Vector Mask类型的命名规则为：

代码块

```
vbool N _t
```

其中，N的取值集合为{1, 2, 4, 8, 16, 32, 64}，示例如下：

```
代码块
1    vbool16_t
```

2.3 Tuple Types

RV Core中，Tuple Types类型的命名规则如下表所示：

Data type	ELEN value	NFIELDS value
v **DTYPE ELEN** mf8 x **NFIELDS** _t	8	2, 3, 4, 5, 6, 7, 8
v **DTYPE ELEN** mf4 x **NFIELDS** _t	8, 16	2, 3, 4, 5, 6, 7, 8
v **DTYPE ELEN** mf2 x **NFIELDS** _t	8, 16, 32	2, 3, 4, 5, 6, 7, 8
v **DTYPE ELEN** m1 x **NFIELDS** _t	8, 16, 32, 64	2, 3, 4, 5, 6, 7, 8
v **DTYPE ELEN** m2 x **NFIELDS** _t	8, 16, 32, 64	2, 3, 4
v **DTYPE ELEN** m4 x **NFIELDS** _t	8, 16, 32, 64	2
v **DTYPE ELEN** m8 x **NFIELDS** _t	X	X

示例如下：

```
代码块
1    vint32m1x3_t，表示可以操作3个vint32m1_t类型的Vector
```

3 常用指令

3.1 配置接口

vsetvl，配置获取VLEN的大小，命名规则如下：

```
代码块
1    vsetvl_SEW_LMUL
2    SEW，元素宽度，一般为e8，e16，e32，e64
3    LMUL，LMUL大小
```

vsetvlmax，获取VLMAX大小，示例如下：

代码块

```
size_t __riscv_vsetvl_e8m1 (size_t avl);
size_t __riscv_vsetvl_e8m2 (size_t avl);
size_t __riscv_vsetvlmax_e8m2();
```

3.2 访存接口

命名规则如下：

代码块

```
__riscv_vl EEW _v_ SEW LMUL [_ TM]    // load func
__riscv_vs EEW _v_ SEW LMUL           // store func
```

示例如下：

代码块

```
float16_t *base, *dest;
size_t vl;
vfloat16mf4_t v1 = __risc_vle16_v_f16mf4(base, vl);
__riscv_vse16_v_f16mf4(dest, v1, vl);
```

3.3 整形接口

命名规则如下：

代码块

```
__riscv_v OP _ OPDS _ SEW LMUL [_ TM]
```

示例如下：

代码块

```
vint32m1_t res = __riscv_vadd_vv_i32m1 (opd1, opd2, vl);
vint32m1_t res = __riscv_vadd_vv_i32m1_m (mask, opd1, opd2, vl);
vuint32m1_t res = __riscv_vadd_vv_u32m1 (opd1, opd2, vl);
vint32m1_t res = __riscv_vsub_vv_i32m1 (opd1, opd2, vl);
vint32m2_t res = __riscv_vwadd_vv_i32m2 (opd1, opd2, vl);
vint32m1_t res = __riscv_vand_vv_i32m1 (opd1, opd2, vl);
```

```
vint32m1_t res = __riscv_vor_vv_i32m1 (opd1, opd2, vl);
vint32m1_t res = __riscv_vxor_vv_i32m1 (opd1, opd2, vl);
vint32m1_t res = __riscv_vnot_v_i32m1 (opd1, vl);
vint32m1_t res = __riscv_vmin_vv_i32m1 (opd1, opd2, vl);
vbool32_t res = __riscv_vmseq_vv_i32m1_b32 (opd1, opd2, vl);
vint32m1_t res = __riscv_vmul_vv_i32m1 (opd1, opd2, vl);

vint32m1_t vd = __riscv_vle32_v_i32m1 (dest, vl);
vd = __riscv_vmacc_vv_i32m1 (vd, opd1, opd2, vl);

vint32m1_t res = __riscv_vdiv_vv_i32m1 (opd1, opd2, vl);
vint32m1_t res = __riscv_vmv_v_v_i32m1 (opd1, vl);
```

3.4 浮点接口

命名规则如下：

代码块

```
__riscv_v OP _ OPDS _ SEW LMUL [_rm] [_ TM]
```

示例如下：

代码块

```
vfloat32m1_t res = __riscv_vfadd_vv_f32m1 (opd1, opd2, vl);
vfloat32m1_t res = __riscv_vfadd_vv_f32m1_m (mask, opd1, opd2, vl);
vfloat32m1_t res = __riscv_vfsub_vv_f32m1 (opd1, opd2, vl);
vfloat32m1_t res = __riscv_vfmul_vv_f32m1 (opd1, opd2, vl);
vfloat32m1_t res = __riscv_vfdiv_vv_f32m1 (opd1, opd2, vl);
vfloat32m1_t res = __riscv_vfrsqrt7_v_f32m1 (opd1, vl);
```

3.5 归并接口

命名规则如下：

代码块

```
__riscv_v OP _vs_ SEW LMUL SEW LMUL [_ TM]
```

示例如下：

代码块

```
1   vint32m1_t res = __riscv_vredsum_vs_i32m1_i32m1 (vector, scalar, vl);
2   vfloat32m1_t res = __riscv_vfredosum_vs_f32m1_f32m1 (vector, scalar, vl);
```

此处需要注意：vredsum为un-ordered的归并，精度没有保证，但运行速度快，vfredosum为ordered 版本，精度有保证但运行速度慢