Static Simulator

Jiaqi Si

CUHK(SZ)

2025-08-01

Outline

1. Architecture Adjustment

1.1 VMACC Instruction Implementation

- Added support for Vector Multiply-Accumulate (VMACC) instructions
- Instruction mapping: `VMACC_VV` ⟶ `FunctionUnitKeyType::VectorMacc`

1.2 Data Forwarding Separation

1.2.1 Common Register Handling

- Delete task queue. Only record the exclusive writing instruction.

```
pub struct CommonRegister { // 8 bytes Register
    pub id : RegisterIdType,
    pub write_instruction : Option<Inst>,
}
```

The handle of vector registers are the same with the previous.

2. Sample Display

2.1 assembly code

```
1021a: 33 8f c7 41    sub t5, a5, t3
1021e: 57 7f 8f 0d    vsetvli t5, t5, e64, m1, ta, ma
10222: 93 1f 3e 00    slli  t6, t3, 0x3
10226: 33 04 f3 01    add s0, t1, t6
1022a: 07 75 04 02    vle64.v v10, (s0)
1022e: f6 9f          add t6, t6, t4
10230: 87 f5 0f 02    vle64.v v11, (t6)
10234: 7a 9e          add t3, t3, t5
10236: d7 94 a5 b2    vfmacc.vv v9, v11, v10
1023a: e3 60 fe fe    bltu  t3, a5, 0x1021a
<matrixmul_intrinsics+0x34>
```

The command : `cargo run -- -i appendix/_matmul/bin/matmul_vector.exe -c ./config.toml -s 0x1021a -e 0x1023a`

2.2 Running log

2.2.1 Data Dependency about scalar instruction

The first four cycles, `sub t5, a5, t3`,`slli  t6, t3, 0x3` and `add s0, t1, t6` are issued. (`vsetvli t5, t5, e64, m1, ta, ma` is ignored)

The log of cycle 2(The third cycle):

```
06:30:09 [INFO] Step 3: Fetching new instructions and
checking if they can be issued
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim: Trying to issue
instruction: Func(FuncInst { raw: ADD { rd: 8, rs1: 6, rs2:
31 }, destination: ScalarRegister(8), resource:
[ScalarRegister(6), ScalarRegister(31)], func_unit_key:
IntegerAlu })
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim: Function
```

2.2 Running log

```
instruction cycles: 1
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim::register:
Register ScalarRegister(31) has unfinished write from
instruction: Func(FuncInst { raw: SLLI { rd: 31, rs1: 28,
shamt: 3 }, destination: ScalarRegister(31), resource:
[ScalarRegister(28)], func_unit_key: IntegerAlu })
```

The `add s0, t1, t6` cannot be issued because `ScalarRegister(31)` is in calculating.

2.2 Running log

2.2.2 Chaining of Vector Instructions

In the cycle 10, `vfmacc.vv v9, v11, v10` is issued.

```
06:30:09 [INFO] Step 3: Fetching new instructions and
checking if they can be issued
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim: Trying to issue
instruction: Func(FuncInst { raw: VMACC_VV { vrd: 9, vrs1:
11, vrs2: 10 }, destination: VectorRegister(9), resource:
[VectorRegister(9), VectorRegister(11), VectorRegister(10)],
func_unit_key: VectorMacc })
```

2.2 Running log

In the cycle 11, register files send data to the input buffer of the function unit `VectorMacc`

```
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim::unit::buffer:
[BufferPair] Current status for FunctionUnit(VectorMacc):
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim::unit::buffer:
[BufferPair] Input buffer resources:
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim::unit::buffer:
[BufferPair]   Resource[0]: Type=Register(VectorRegister(9)),
Progress=32/128 bytes (25.00%)
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim::unit::buffer:
[BufferPair]   Resource[1]:
Type=Register(VectorRegister(11)), Progress=32/128 bytes
(25.00%)
```

2.2 Running log

```
06:30:09 [DEBUG] (1) ruscv_vector_sim::sim::unit::buffer:
[BufferPair]   Resource[2]:
Type=Register(VectorRegister(10)), Progress=32/128 bytes
(25.00%)
```

2.2 Running log

In the same cycle, `vle64.v v11, (t6)` is still in working

```
04:52:51 [DEBUG] (1) ruscv_vector_sim::sim::unit::buffer:
[BufferPair] Increasing result buffer by 32 bytes for
MemoryUnit(Load(1))
04:52:51 [DEBUG] (1) ruscv_vector_sim::sim::unit::buffer:
[BufferPair] Result buffer after increase: Current=96/128
bytes (75.00%), Total processed=64/128 bytes (50.00%)
04:52:51 [DEBUG] (1)
ruscv_vector_sim::sim::unit::memory_unit: Read port 1 task
not completed: current_pos=96/128 bytes, result buffer
completed=false
```