⚙ Differential > D69987

⚙ [RISCV] Assemble/Disassemble v-ext instructions.

</> Needs Review 🌐 Public

≡ Actions

Authored by **HsiangKai** on Nov 7 2019, 10:43 PM.

Tags

LLVM

clang

Subscribers

cfe-commits, evandro, Joy12138 and 28 others

Details

Reviewers

- ○ asb
- 💬 rogfer01
- ○ rkruppe
- 💬 kito-cheng
- ○ khchen
- 💬 evandro

≡ SUMMARY

Assemble/disassemble RISC-V V extension instructions according to version 0.8-draft-20191004 in https://github.com/riscv/riscv-v-spec/.

I have tested this patch using GNU toolchain. The encoding is aligned to GNU assembler output. In this patch, there is a test case for each instruction at least.

The V register definition is just for assemble/disassemble. Its type is not important in this stage. I think it will be reviewed and modified as we want to do codegen for scalable vector types.

This patch does not include Zvamo, Zvlsseg, and Zvediv.

Diff Detail

Repository

rG LLVM Github Monorepo

Build Status

- ✖ **Buildable 53851**
- ✖ Build 61570: pre-merge checks

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

hanges, so older changes are hidden. Show Older Changes

evandro added a comment. Feb 4 2020, 4:50 PM

Shouldn't most instructions have the `vtype` register in `Uses` ? Shouldn't most FP instructions have `fflags` in `Defs` ?

The tests are extensive, but it would be important to also add negative tests.

llvm/lib/Target/RISCV/AsmParser/RISCVAsmParser.cpp

158	Please, rename to `parseVTypeI()` .
208	Please, rename to `defaultMaskRegOp()` .
755	This should be the `default` case for the `switch` above.
769	Ditto.
851	Avoid calling `logBase2()` multiple times.
852	Ditto.
1163	Please, rename to `Match_InvalidVTypeI` .
1168	Please, rename to `Match_InvalidVMaskRegister` .
1522	Please, rename to `parseVTypeI()` .
1565	Please, rename to `parseMaskReg()` .

llvm/lib/Target/RISCV/Disassembler/RISCVDisassembler.cpp

165	Please, rename to `decodeVMaskReg()` .

llvm/lib/Target/RISCV/MCTargetDesc/RISCVInstPrinter.cpp

164	Please, rename to `printVMaskReg()` .

llvm/lib/Target/RISCV/MCTargetDesc/RISCVInstPrinter.h

46	Please, rename to `printVMaskReg()` .

llvm/lib/Target/RISCV/MCTargetDesc/RISCVMCCodeEmitter.cpp

84	Please, rename to `getVMaskReg()` .

llvm/lib/Target/RISCV/RISCV.td

152	Should it also imply `FeatureStdExtD` ?

llvm/lib/Target/RISCV/RISCVInstrInfoV.td

33	Please, rename to `VRegAsmOperand` .
40	Please, rename to `VRegOp` .
	`smOperand` .
	`p` .

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

78 No need to set fields to the default `0` .

535 Shouldn't these have `vxsat` in `Defs` ?

541 Shouldn't these have `vxrm` in `Uses` ?

547 Shouldn't this hace `vxrm` in `Uses` and `vxsat` in `Defs` ?

550 Aren't these part of the base V extension?

556 Shouldn't this hace `vxrm` in `Uses` ?

560 Shouldn't this hace `vxrm` in `Uses` and `vxsat` in `Defs` ?

HsiangKai updated this revision to **Diff 242816**. Edited · Feb 6 2020, 12:17 AM

HsiangKai marked 23 inline comments as done.

Address **@fpallares** and **@evandro** 's comments.

About `fflags` in FP instructions, it is also related to RISCVInstrInfoF.td and RISCVInstrInfoD.td. So, I think we could prepare another patch for it.

HsiangKai added inline comments. Feb 6 2020, 12:18 AM

llvm/lib/Target/RISCV/RISCV.td

152 I didn't find that V will imply D extension. Could you indicate the description in spec about this?

llvm/lib/Target/RISCV/RISCVInstrInfoV.td

78 There is no default value for hasSideEffects, mayLoad, and mayStore.

535 It should have no impact on CodeGen. It seems no instructions to use vxsat.

HsiangKai updated this revision to **Diff 242905**. Feb 6 2020, 7:55 AM

Add the predicate for extension Zvqmac.

simoncook mentioned this in ~~**D73891: [RISCV] Support experimental/unratified extensions**~~.

Feb 7 2020, 2:44 AM

fpallares added a comment. Feb 14 2020, 8:46 AM

Thanks for addressing my comments **@HsiangKai** . After a closer look I've added some more inline comments, and some general ones too:

ove instructions `vmv<nf>r.v` .

that are missing the register overlap constraints (see inline

`smParser::validateInstruction` , as well as new tests.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

- I've noticed you are using `${vm}` for describing the mask arg on most instructions, but AFAICT `$vm` is equally valid (e.g. You could write `$vd, $vs2$vm` instead of `$vd, $vs2${vm}`).
- We should probably recognize the assembler pseudo instructions defined in the ISA. Most can be implemented via `InstAlias` . I suggest to do this in another patch.
- Instruction from the `Zvqmac` extension instructions were added, should we add instructions for the other extensions as well? Otherwise we might want to restrict this patch to the base only.

llvm/lib/Target/RISCV/Disassembler/RISCVDisassembler.cpp

↪ 154 Following **@evandro** 's naming suggestions, this should probably be renamed to `DecodeVRegisterClass` .

llvm/lib/Target/RISCV/RISCVInstrInfoV.td

↪ 40 Whitespace missing before the colon.

↪ 55 Ditto.

↪ 107 I don't see why setting `hasSideEffects` is necessary and setting `mayStore` is not enough, could you please elaborate a bit on this?

↪ 426 Whitespace missing before the colon.

↪ 438 I'd say those instructions need the `@earlyclobber $vd` restriction too. Not because of the first source operand (since it has the same width as the destination), but because of the second (in the case of `VWOP_WV`), or the mask (for `VWOP_WX`).

Quoting section 11.2. Widening Vector Arithmetic Instructions:

> *The destination vector register group cannot overlap a source vector register group of a different element width (including the mask register if masked), otherwise an illegal instruction exception is raised.*

This has the downside that the `earlyclobber` constraint is too coarse and will impose unnecessary restrictions by not allowing the destination to overlap with the first (wide) operand, even when the instruction is used unmasked. Maybe we can use the `earlyclobber` for now, and find a better solution for this in a later patch.

↪ 460 In section 11.3. Narrowing Vector Arithmetic Instructions we find the following paragraph:

> *The destination vector register group cannot overlap the first source vector register group (specified by vs2). The destination vector register group cannot overlap the mask register if used, unless LMUL=1. If either constraint is violated, an illegal instruction exception is raised.*

From my understanding, this applies to `VNSRL` and `VNSRA` . In that case we need an `@earlyclobber` constraint here.

↪ 480 Whitespace missing before the colon.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

on narrowing instructions apply to `VNCLIP` and `VNCLIPU` too.

ine comment above (for `VWADD_W` and such), there is an overlap restriction for these instructions too.

584 Whitespace missing before the colon.

661 I believe we are also missing the `@earlyclobber $vd` overlap constraints for these narrowing conversions.

From section 14.17. Narrowing Floating-Point/Integer Type-Convert Instructions:

> *These instructions have the same constraints on vector register overlap as other narrowing instructions (see Narrowing Vector Arithmetic Instructions).*

675 If I'm not mistaken, those instructions should have the `@earlyclobber $vd` constraint too to avoid overlaps.

685 Ditto.

llvm/lib/Target/RISCV/RISCVRegisterInfo.td

50 My understanding is that here you specify minimum sizes (at least we will have one element of size `ELEN`). As they stand, they are consistent with `ELEN=64` . So far it is unclear to me what sizes we would want to use if we also want to support `ELEN=32` . I imagine we could reuse `HwMode` or something similar.

As far as the offsets are concerned I'd suggest setting them to `-1` because we don't really know the offset where the `hi` part starts.

That said, you already mentioned that types are not that important in this patch.

llvm/test/MC/RISCV/rvv/macc.s

2 Looks like `+v` attribute gets passed twice. I'm also not sure we need `+a` and `+c` .

evandro added inline comments. Feb 18 2020, 9:22 AM

llvm/lib/Target/RISCV/RISCV.td

152 Indeed it doesn't:

> *If the base scalar ISA does not include floating-point, then a `fcsr` register is also added to hold mirrors of the `vxsat` and `vxrm` CSRs as explained below.*

Which means that the instructions that take FP scalars should also depend on `hasStdExt{F|D}` . Likewise when the scalar is a 64 bit integer, `hasRV64` .

llvm/lib/Target/RISCV/RISCVInstrInfoV.td

78 Right. It would be cleaner if they were moved to the respective base instruction class then. Just a suggestion.

107 `hasSideEffects` is not necessary, since `mayStore` is indeed enough. The purpose of [illegible] catch all when the rest of the instruction attributes miss some side

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

535 For the sake of completeness.

rogfer01 added inline comments. Feb 18 2020, 1:00 PM

llvm/lib/Target/RISCV/RISCV.td

152 — I don't think we have to do anything special with 64 bit: section 11.1 states what is the behaviour when `XLEN` is different than `SEW`.

evandro added inline comments. Feb 18 2020, 1:24 PM

llvm/lib/Target/RISCV/RISCV.td

152 — Good point. 👍

HsiangKai marked 3 inline comments as done. Feb 19 2020, 12:09 AM

HsiangKai added inline comments.

llvm/lib/Target/RISCV/Disassembler/RISCVDisassembler.cpp

154 — The function name will be expected by TableGen as `"Decode" + Record->getName().str() + "RegisterClass"`. It seems related to the name of register class. Currently, the name of RegisterClass for V-extension registers is "VR".

llvm/lib/Target/RISCV/RISCVInstrInfoV.td

107 — There is no default value for hasSideEffects, mayLoad, and mayStore in `Instruction` class. So, I specify these three values for V instructions. It follows the style in RISCVInstrInfo.td. I agree that we could give these attributes some default values in some base class for V instructions. I will try to refactor it.

460 — In current implementation for MC layer, there is no LMUL information in these instructions. So, if LMUL=1, there is no such restriction for them. I have a downstream patch to extend these instruction definitions for LMUL > 1. Maybe we could keep this MC implementation for LMUL=1 and extend it later. I will add comments for it.

HsiangKai updated this revision to **Diff 245339**. Feb 19 2020, 12:19 AM

Address some of @fpallares' comments.

HsiangKai updated this revision to **Diff 245341**. Feb 19 2020, 12:37 AM

Remove Zvqmac extension. Keep this patch for base only.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

2020, 12:47 AM

llvm/lib/Target/RISCV/RISCVInstrInfoV.td

↪ 460

Perhaps the wording in the spec is unclear here?

Looks like the following sentence does apply regardless of LMUL.

> > *The destination vector register group cannot overlap the first source vector register group (specified by vs2).*

While the sentence that follows it is clear that applies only to LMUL>1

> > *The destination vector register group cannot overlap the mask register if used, unless LMUL=1*

You also said that

> *I have a downstream patch to extend these instruction definitions for LMUL > 1*

I'm curious here: did you use a Pseudo for that or somehow avoided the issue that they'd be encoded in the same way?

Thanks!

fpallares marked an inline comment as done. Feb 19 2020, 2:23 AM

fpallares added inline comments.

llvm/lib/Target/RISCV/Disassembler/RISCVDisassembler.cpp

↪ 154

I see, sorry for the noise.

llvm/lib/Target/RISCV/RISCVInstrInfoV.td

↪ 107

I was only referring tot he fact that `hasSideEffects` here changed from `0` to `1` in a recent update of the patch. I don't really think a refactor is necessary here.

HsiangKai marked 2 inline comments as done. Feb 19 2020, 7:33 PM

HsiangKai added inline comments.

llvm/lib/Target/RISCV/RISCV.td

↪ 152

Yes, it is a problem about accessing fcsr if F extension is turned off. However, I found that there is vcsr in the latest version.

https://github.com/riscv/riscv-v-spec/commit/b25b643b9c29b4fde570293c64ca682a99a09f2a

So, I think we could keep V and F as separate unrelated extensions.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

ctor floating point operations.

td

460

I misunderstood the statements. Sorry for that. I will add earlyclobber for narrowing instructions. Thanks a lot.

About instruction definitions for LMUL > 1, I enable 'isCodeGenOnly' for these instructions to avoid encoding checking.

HsiangKai updated this revision to **Diff 245568**. Feb 19 2020, 7:38 PM

- Add predicates for vector floating-point instructions.
- Add earlyclobber for narrowing instructions.

HsiangKai updated this revision to **Diff 245571**. Feb 19 2020, 8:40 PM

HsiangKai updated this revision to **Diff 245577**. Feb 19 2020, 10:01 PM

Update test cases.

HsiangKai updated this revision to **Diff 245579**. Feb 19 2020, 10:18 PM

fpallares added inline comments. Mar 6 2020, 2:20 AM

llvm/test/MC/RISCV/rvv/invalid.s

6

Suggestion: If you use `CHECK-LABEL` between the regular `CHECK` you can avoid these directives to succeed by matching lines corresponding to a later instruction. Like this:

```
vsetvli a2, a0, e31
// CHECK-ERROR: operand must be e[8|16|32|64|128|256|512|1024],m[1|2|4|8]
// CHECK-ERROR-LABEL: vsetvli a2, a0, e31
```

(Note the `CHECK-LABEL` is placed *after* the `CHECK` since it appears in this order in the output)

Without the `CHECK-LABELS` the test will still fail when something is wrong, but with them FileCheck will be able to point exactly where the problem occurred.

More info on this in the corresponding section of the FileCheck manual.

HsiangKai marked an inline comment as done. Mar 10 2020, 8:43 AM

HsiangKai added inline comments.

llvm/test/MC/RISCV/rvv/invalid.s

6

Got it. Thanks.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

49399. Mar 10 2020, 8:45 AM

In RISCVRegisterInfo.td, set offset of vector sub-registers to -1.

Harbormaster failed remote builds in **B48696: Diff 249399**! Mar 10 2020, 9:46 AM

HsiangKai updated this revision to **Diff 249833**. Mar 11 2020, 10:18 PM

- Rewrite validateInstruction() in RISCVAsmParser.cpp. Use TSFlags to classify instructions' constraints.
- Update invalid.s test cases.

Harbormaster failed remote builds in **B48928: Diff 249833**! Mar 11 2020, 11:18 PM

HsiangKai updated this revision to **Diff 254501**. Thu, Apr 2, 6:27 AM

- Rebase on master branch.
- Fix validInstruction() bugs.
- Update test cases.

Harbormaster failed remote builds in **B51467: Diff 254501**! Thu, Apr 2, 7:01 AM

HsiangKai updated this revision to **Diff 254667**. Thu, Apr 2, 6:23 PM

- Fix clang-format errors.
- Rename variables according to LLVM Coding Standards.

Harbormaster failed remote builds in **B51573: Diff 254667**! Thu, Apr 2, 7:00 PM

HsiangKai updated this revision to **Diff 258526**. Sat, Apr 18, 8:48 AM

- Rebase master.
- Make "V" extension as an experimental one.
  - This follows the design as discussed on the mailing lists in the following RFC: http://lists.llvm.org/pipermail/llvm-dev/2020-January/138364.html

Herald added a project: **clang**. · View Herald Transcript Sat, Apr 18, 8:48 AM

Herald added a subscriber: **cfe-commits**. · View Herald Transcript

Harbormaster failed remote builds in **B53851: Diff 258526**! Sat, Apr 18, 9:09 AM

Revision Contents ≣ **Changeset List**

Files History Commits Similar

Path **Packages**

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

V.cpp (5 lines)

M llvm/lib/Target/RISCV/AsmParser/RISCVAsmParser.cpp (258 lines)

M llvm/lib/Target/RISCV/Disassembler/RISCVDisassembler.cpp (36 lines)

M llvm/lib/Target/RISCV/MCTargetDesc/RISCVInstPrinter.h (4 lines)

M llvm/lib/Target/RISCV/MCTargetDesc/RISCVInstPrinter.cpp (24 lines)

M llvm/lib/Target/RISCV/MCTargetDesc/RISCVMCCodeEmitter.cpp (20 lines)

M llvm/lib/Target/RISCV/RISCV.td (7 lines)

M llvm/lib/Target/RISCV/RISCVInstrFormats.td (19 lines)

A M llvm/lib/Target/RISCV/RISCVInstrFormatsV.td (300 lines)

M llvm/lib/Target/RISCV/RISCVInstrInfo.h (22 lines)

M llvm/lib/Target/RISCV/RISCVInstrInfo.td (1 line)

A M llvm/lib/Target/RISCV/RISCVInstrInfoV.td (820 lines)

M llvm/lib/Target/RISCV/RISCVRegisterInfo.td (104 lines)

M llvm/lib/Target/RISCV/RISCVSchedRocket32.td (1 line)

M llvm/lib/Target/RISCV/RISCVSchedRocket64.td (1 line)

M llvm/lib/Target/RISCV/RISCVSubtarget.h (2 lines)

M llvm/lib/Target/RISCV/RISCVSystemOperands.td (10 lines)

M llvm/lib/Target/RISCV/Utils/RISCVBaseInfo.h (2 lines)

A M llvm/test/MC/RISCV/rvv/add.s (315 lines)

A M llvm/test/MC/RISCV/rvv/and.s (45 lines)

A M llvm/test/MC/RISCV/rvv/clip.s (81 lines)

A M llvm/test/MC/RISCV/rvv/compare.s (249 lines)

A M llvm/test/MC/RISCV/rvv/convert.s (189 lines)

A M llvm/test/MC/RISCV/rvv/div.s (105 lines)

A M llvm/test/MC/RISCV/rvv/fadd.s (81 lines)

A M llvm/test/MC/RISCV/rvv/fcompare.s (129 lines)

A M llvm/test/MC/RISCV/rvv/fdiv.s (45 lines)

A M llvm/test/MC/RISCV/rvv/fmacc.s (297 lines)

A M llvm/test/MC/RISCV/rvv/fminmax.s (57 lines)

A M llvm/test/MC/RISCV/rvv/fmul.s (57 lines)

s)

ines)

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

A M llvm/test/MC/RISCV/rvv/freduction.s (81 lines)

A M llvm/test/MC/RISCV/rvv/fsub.s (93 lines)

A M llvm/test/MC/RISCV/rvv/invalid.s (550 lines)

A M llvm/test/MC/RISCV/rvv/load.s (339 lines)

A M llvm/test/MC/RISCV/rvv/macc.s (189 lines)

A M llvm/test/MC/RISCV/rvv/mask.s (141 lines)

A M llvm/test/MC/RISCV/rvv/minmax.s (105 lines)

A M llvm/test/MC/RISCV/rvv/mul.s (201 lines)

A M llvm/test/MC/RISCV/rvv/mv.s (63 lines)

A M llvm/test/MC/RISCV/rvv/or.s (45 lines)

A M llvm/test/MC/RISCV/rvv/others.s (141 lines)

A M llvm/test/MC/RISCV/rvv/reduction.s (129 lines)

A M llvm/test/MC/RISCV/rvv/shift.s (261 lines)

A M llvm/test/MC/RISCV/rvv/sign-injection.s (81 lines)

A M llvm/test/MC/RISCV/rvv/snippet.s (32 lines)

A M llvm/test/MC/RISCV/rvv/store.s (207 lines)

A M llvm/test/MC/RISCV/rvv/sub.s (285 lines)

A M llvm/test/MC/RISCV/rvv/vsetvl.s (21 lines)

A M llvm/test/MC/RISCV/rvv/xor.s (45 lines)

Diff 258526

clang/lib/Driver/ToolChains/Arch/RISCV.cpp

☰ View Options

Show First 20 Lines • Show All 56 Lines • ▼ Show 20 Line(s)

```
// If the extension is supported as experimental, return the version of that
// extension that the compiler currently supports.
static Optional<RISCVExtensionVersion>
isExperimentalExtension(StringRef Ext) {
  if (Ext == "b" || Ext == "zbb" || Ext == "zbc" || Ext == "zbe" ||
      Ext == "zbf" || Ext == "zbm" || Ext == "zbp" || Ext == "zbr" ||
      Ext == "zbs" || Ext == "zbt" || Ext == "zbproposedc")
    return RISCVExtensionVersion{"0", "92"};
  if (Ext == "v")
    return RISCVExtensionVersion{"0", "8"};
  return None;
}

                                      n(StringRef Ext) {
                                      ns which are marked as experimental.
                                      xt))
    return true;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

▲ Show 20 Lines • Show All 321 Lines • ▼ Show 20 Line(s)

```
      HasD = true;
      break;
    case 'c':
      Features.push_back("+c");
      break;
    case 'b':
      Features.push_back("+experimental-b");
      break;
    case 'v':
      Features.push_back("+experimental-v");
      break;
    }

    // Consume full extension name and version, including any optional '_'
    // between this extension and the next
    ++I;
    I += Major.size();
    if (Minor.size())
      I += Minor.size() + 1 /*'p'*/;
```

▲ Show 20 Lines • Show All 248 Lines • Show Last 20 Lines

clang/test/Driver/riscv-arch.c

≡ View Options

Show First 20 Lines • Show All 354 Lines • ▼ Show 20 Line(s)

```
// RUN: %clang -target riscv32-unknown-elf -march=rv32izbb0p92 -menable-experimental-extensions -### %s \
// RUN: -fsyntax-only 2>&1 | FileCheck -check-prefix=RV32-EXPERIMENTAL-ZBB %s
// RV32-EXPERIMENTAL-ZBB: "-target-feature" "+experimental-zbb"

// RUN: %clang -target riscv32-unknown-elf -march=rv32izbb0p92_zbp0p92 -menable-experimental-extensions -### %s \
// RUN: -fsyntax-only 2>&1 | FileCheck -check-prefix=RV32-EXPERIMENTAL-ZBB-ZBP %s
// RV32-EXPERIMENTAL-ZBB-ZBP: "-target-feature" "+experimental-zbb"
// RV32-EXPERIMENTAL-ZBB-ZBP: "-target-feature" "+experimental-zbp"

// RUN: %clang -target riscv32-unknown-elf -march=rv32iv -### %s \
// RUN: -fsyntax-only 2>&1 | FileCheck -check-prefix=RV32-EXPERIMENTAL-V-NOFLAG %s
// RV32-EXPERIMENTAL-V-NOFLAG: error: invalid arch name 'rv32iv'
// RV32-EXPERIMENTAL-V-NOFLAG: requires '-menable-experimental-extensions'

// RUN: %clang -target riscv32-unknown-elf -march=rv32iv -menable-experimental-extensions -### %s \
// RUN: -fsyntax-only 2>&1 | FileCheck -check-prefix=RV32-EXPERIMENTAL-V-NOVERS %s
// RV32-EXPERIMENTAL-V-NOVERS: error: invalid arch name 'rv32iv'
// RV32-EXPERIMENTAL-V-NOVERS: experimental extension requires explicit version number

// RUN: %clang -target riscv32-unknown-elf -march=rv32iv0p1 -menable-experimental-extensions -### %s \
// RUN: -fsyntax-only 2>&1 | FileCheck -check-prefix=RV32-EXPERIMENTAL-V-BADVERS %s
// RV32-EXPERIMENTAL-V-BADVERS: error: invalid arch name 'rv32iv0p1'
// RV32-EXPERIMENTAL-V-BADVERS: unsupported version number 0.1 for experimental extension

unknown-elf -march=rv32iv0p8 -menable-experimental-extensions
leCheck -check-prefix=RV32-EXPERIMENTAL-V-GOODVERS %s
: "-target-feature" "+experimental-v"
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

llvm/lib/Target/RISCV/AsmParser/RISCVAsmParser.cpp

≡ View Options

```
//===-- RISCVAsmParser.cpp - Parse RISCV assembly to MCInst instructions --===//
//
// Part of the LLVM Project, under the Apache License v2.0 with LLVM Exceptions.
// See https://llvm.org/LICENSE.txt for license information.
// SPDX-License-Identifier: Apache-2.0 WITH LLVM-exception
//
//===----------------------------------------------------------------------===//

#include "MCTargetDesc/RISCVAsmBackend.h"
#include "MCTargetDesc/RISCVMCExpr.h"
#include "MCTargetDesc/RISCVMCTargetDesc.h"
#include "MCTargetDesc/RISCVTargetStreamer.h"
#include "RISCVInstrInfo.h"
#include "TargetInfo/RISCVTargetInfo.h"
#include "Utils/RISCVBaseInfo.h"
#include "Utils/RISCVMatInt.h"
#include "llvm/ADT/STLExtras.h"
#include "llvm/ADT/SmallBitVector.h"
#include "llvm/ADT/SmallVector.h"
#include "llvm/ADT/Statistic.h"
#include "llvm/ADT/StringSwitch.h"
#include "llvm/CodeGen/Register.h"
#include "llvm/MC/MCAssembler.h"
#include "llvm/MC/MCContext.h"
#include "llvm/MC/MCExpr.h"
#include "llvm/MC/MCInst.h"
```

▲ Show 20 Lines • Show All 98 Lines • ▼ Show 20 Line(s)

```
                           MCStreamer &Out, bool HasTmpReg);

  // Checks that a PseudoAddTPRel is using x4/tp in its second input operand.
  // Enforcing this using a restricted register class for the second input
  // operand of PseudoAddTPRel results in a poor diagnostic due to the fact
  // 'add' is an overloaded mnemonic.
  bool checkPseudoAddTPRel(MCInst &Inst, OperandVector &Operands);

  // Check instruction constraints.
  bool validateInstruction(MCInst &Inst, OperandVector &Operands);

  /// Helper for processing MC instructions that have been successfully matched
  /// by MatchAndEmitInstruction. Modifications to the emitted instructions,
  /// like the expansion of pseudo instructions (e.g., "li"), can be performed
  /// in this method.
  bool processInstruction(MCInst &Inst, SMLoc IDLoc, OperandVector &Operands,
                          MCStreamer &Out);

// Auto-generated instruction matching functions
#define GET_ASSEMBLER_HEADER
#include "RISCVGenAsmMatcher.inc"

  OperandMatchResultTy parseCSRSystemRegister(OperandVector &Operands);
  OperandMatchResultTy parseImmediate(OperandVector &Operands);
  OperandMatchResultTy parseRegister(OperandVector &Operands,
                                     bool AllowParens = false);
                               OpBaseReg(OperandVector &Operands);
                               micMemOp(OperandVector &Operands);
                               randWithModifier(OperandVector &Operands);
  OperandMatchResultTy parseBareSymbol(OperandVector &Operands);
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
150  155    OperandMatchResultTy parseCallSymbol(OperandVector &Operands);
151  156    OperandMatchResultTy parsePseudoJumpSymbol(OperandVector &Operands);
152  157    OperandMatchResultTy parseJALOffset(OperandVector &Operands);
     158    OperandMatchResultTy parseVTypeI(OperandVector &Operands);
```

evandro ✔ Done

Please, rename to `parseVTypeI()` .

```
     159    OperandMatchResultTy parseMaskReg(OperandVector &Operands);
153  160
154  161    bool parseOperand(OperandVector &Operands, StringRef Mnemonic);
155  162
156  163    bool parseDirectiveOption();
157  164    bool parseDirectiveAttribute();
158  165
159  166    void setFeatureBits(uint64_t Feature, StringRef FeatureString) {
160  167      if (!(getSTI().getFeatureBits()[Feature])) {
```

Show All 32 Lines

```
193  200      copySTI().setFeatureBits(FeatureBits);
194  201      setAvailableFeatures(ComputeAvailableFeatures(FeatureBits));
195  202
196  203      ParserOptions = ParserOptionsStack.pop_back_val();
197  204
198  205      return false;
199  206    }
200  207
     208    std::unique_ptr<RISCVOperand> defaultMaskRegOp() const;
```

evandro ✔ Done

Please, rename to `defaultMaskRegOp()` .

```
     209
201  210  public:
202  211    enum RISCVMatchResultTy {
203  212      Match_Dummy = FIRST_TARGET_MATCH_RESULT_TY,
204  213  #define GET_OPERAND_DIAGNOSTIC_TYPES
205  214  #include "RISCVGenAsmMatcher.inc"
206  215  #undef GET_OPERAND_DIAGNOSTIC_TYPES
207  216    };
208  217
```

Show All 31 Lines

```
240  249  /// RISCVOperand - Instances of this class represent a parsed machine
241  250  /// instruction
242  251  struct RISCVOperand : public MCParsedAsmOperand {
243  252
244  253    enum class KindTy {
245  254      Token,
246  255      Register,
247  256      Immediate,
248           SystemRegister
     257      SystemRegister,
     258      VType,
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
253  263    struct RegOp {
    Register RegNum;
  };

  struct ImmOp {
    const MCExpr *Val;
  };

  struct SysRegOp {
    const char *Data;
    unsigned Length;
    unsigned Encoding;
    // FIXME: Add the Encoding parsed fields as needed for checks,
    // e.g.: read/write or user/supervisor/machine privileges.
  };

  enum class VSEW {
    SEW_8 = 0,
    SEW_16,
    SEW_32,
    SEW_64,
    SEW_128,
    SEW_256,
    SEW_512,
    SEW_1024,
  };

  enum class VLMUL { LMUL_1 = 0, LMUL_2, LMUL_4, LMUL_8 };

  struct VTypeOp {
    VSEW Sew;
    VLMUL Lmul;
    unsigned Encoding;
  };

```

rogfer01 Not Done

You will have to replace this with another one `VType` otherwise clang (at least 7.0) will reject this code. I suggest `VTypeOp` .

```
  SMLoc StartLoc, EndLoc;
  union {
    StringRef Tok;
    RegOp Reg;
    ImmOp Imm;
    struct SysRegOp SysReg;
    struct VTypeOp VType;
```

rogfer01 Not Done

`VTypeOp VType;` should do here

```
  };

  RISCVOperand(KindTy K) : MCParsedAsmOperand(), Kind(K) {}

nd &o) : MCParsedAsmOperand() {

    IsRV64 = o.IsRV64;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
    StartLoc = o.StartLoc;
    EndLoc = o.EndLoc;
    switch (Kind) {
    case KindTy::Register:
      Reg = o.Reg;
      break;
    case KindTy::Immediate:
      Imm = o.Imm;
      break;
    case KindTy::Token:
      Tok = o.Tok;
      break;
    case KindTy::SystemRegister:
      SysReg = o.SysReg;
      break;
    case KindTy::VType:
      VType = o.VType;
      break;
    }
  }

  bool isToken() const override { return Kind == KindTy::Token; }
  bool isReg() const override { return Kind == KindTy::Register; }
  bool isV0Reg() const {
    return Kind == KindTy::Register && Reg.RegNum == RISCV::V0;
  }
  bool isImm() const override { return Kind == KindTy::Immediate; }
  bool isMem() const override { return false; }
  bool isSystemRegister() const { return Kind == KindTy::SystemRegister; }
  bool isVType() const { return Kind == KindTy::VType; }

  bool isGPR() const {
    return Kind == KindTy::Register &&
           RISCVMCRegisterClasses[RISCV::GPRRegClassID].contains(Reg.RegNum);
  }

  static bool evaluateConstantImm(const MCExpr *Expr, int64_t &Imm,
                                  RISCVMCExpr::VariantKind &VK) {
```

▲ Show 20 Lines • Show All 67 Lines • ▼ Show 20 Line(s)

```
    if (!isImm() || evaluateConstantImm(getImm(), Imm, VK))
      return false;
    return RISCVAsmParser::classifySymbolRef(getImm(), VK, Imm) &&
           VK == RISCVMCExpr::VK_RISCV_TPREL_ADD;
  }

  bool isCSRSystemRegister() const { return isSystemRegister(); }

  bool isVTypeI() const { return isVType(); }

  /// Return true if the operand is a valid for the fence instruction e.g.
  /// ('iorw').
  bool isFenceArg() const {
    if (!isImm())
      return false;
    const MCExpr *Val = getImm();
    auto *SVal = dyn_cast<MCSymbolRefExpr>(Val);
                                 ) != MCSymbolRefExpr::VK_None)
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

s • Show All 91 Lines • ▼ Show 20 Line(s)

```
                                VK = RISCVMCExpr::VK_RISCV_None;
    if (!isImm())
```

```
      return false;
    bool IsConstantImm = evaluateConstantImm(getImm(), Imm, VK);
    return IsConstantImm && isUInt<5>(Imm) && (Imm != 0) &&
           VK == RISCVMCExpr::VK_RISCV_None;
  }

  bool isSImm5() const {
    if (!isImm())
      return false;
    RISCVMCExpr::VariantKind VK = RISCVMCExpr::VK_RISCV_None;
    int64_t Imm;
    bool IsConstantImm = evaluateConstantImm(getImm(), Imm, VK);
    return IsConstantImm && isInt<5>(Imm) && VK == RISCVMCExpr::VK_RISCV_None;
  }

  bool isSImm6() const {
    if (!isImm())
      return false;
    RISCVMCExpr::VariantKind VK = RISCVMCExpr::VK_RISCV_None;
    int64_t Imm;
    bool IsConstantImm = evaluateConstantImm(getImm(), Imm, VK);
    return IsConstantImm && isInt<6>(Imm) &&
      VK == RISCVMCExpr::VK_RISCV_None;
```

▲ Show 20 Lines • Show All 176 Lines • ▼ Show 20 Line(s)

```
    return Imm.Val;
  }

  StringRef getToken() const {
    assert(Kind == KindTy::Token && "Invalid type access!");
    return Tok;
  }

  static StringRef getSEWStr(VSEW Sew) {
    switch (Sew) {
    default:
```

rogfer01 Not Done

This label is unneeded because this switch is covering all of them.

As per the llvm coding standards you dont't want a default in this case.

Also the formatting is unusal, make sure you run clang-format to the new code you contribute.

```
      llvm_unreachable("SEW must be [8|16|32|64|128|256|512|1024]");
    case VSEW::SEW_8:
      return "e8";
    case VSEW::SEW_16:
      return "e16";
    case VSEW::SEW_32:
      return "e32";
    case VSEW::SEW_64:
      return "e64";
    case VSEW::SEW_128:
      return "e128";
```

```
    case VSEW::SEW_1024:
      return "e1024";
```

evandro ✔ Done

This should be the `default` case for the `switch` above.

```
    }
  }

  static StringRef getLMULStr(VLMUL Lmul) {
    switch (Lmul) {
```

rogfer01 Not Done

Ditto.

```
    default:
      llvm_unreachable("LMUL must be [1|2|4|8]");
    case VLMUL::LMUL_1:
      return "m1";
    case VLMUL::LMUL_2:
      return "m2";
    case VLMUL::LMUL_4:
      return "m4";
    case VLMUL::LMUL_8:
```

evandro ✔ Done

Ditto.

```
      return "m8";
    }
  }
```

kito-cheng Not Done

Indention seems weird for getSEWStr and getLMULStr?

```

  StringRef getVType(SmallVectorImpl<char> &Out) const {
    assert(Kind == KindTy::VType && "Invalid access!");
    Twine VTypeStr(getSEWStr(VType.Sew));
    VTypeStr.concat(Twine(","));
    VTypeStr.concat(Twine(getLMULStr(VType.Lmul)));

    return VTypeStr.toStringRef(Out);
  }

  void print(raw_ostream &OS) const override {
    switch (Kind) {
    case KindTy::Immediate:
      OS << *getImm();
      break;
    case KindTy::Register:
      OS << "<register x";
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
< "'";
      break;
```

```
    case KindTy::SystemRegister:
      OS << "<sysreg: " << getSysReg() << '>';
      break;
    case KindTy::VType:
      SmallVector<char, 8> VTypeBuf;
      OS << "<vtype: " << getVType(VTypeBuf) << '>';
      break;
    }
  }

  static std::unique_ptr<RISCVOperand> createToken(StringRef Str, SMLoc S,
                                                   bool IsRV64) {
    auto Op = std::make_unique<RISCVOperand>(KindTy::Token);
    Op->Tok = Str;
    Op->StartLoc = S;
```

Show All 28 Lines

```
    Op->SysReg.Data = Str.data();
    Op->SysReg.Length = Str.size();
    Op->SysReg.Encoding = Encoding;
    Op->StartLoc = S;
    Op->IsRV64 = IsRV64;
    return Op;
  }

  static std::unique_ptr<RISCVOperand> createVType(APInt Sew, APInt Lmul,
                                                   SMLoc S, bool IsRV64) {
    auto Op = std::make_unique<RISCVOperand>(KindTy::VType);
    Sew.ashrInPlace(3);
    unsigned SewLog2 = Sew.logBase2();
    unsigned LmulLog2 = Lmul.logBase2();
```

evandro ✔ Done

Avoid calling `logBase2()` multiple times.

```
    Op->VType.Sew = static_cast<VSEW>(SewLog2);
```

evandro ✔ Done

Ditto.

```
    Op->VType.Lmul = static_cast<VLMUL>(LmulLog2);
    Op->VType.Encoding = (SewLog2 << 2) | LmulLog2;
    Op->StartLoc = S;
    Op->IsRV64 = IsRV64;
    return Op;
  }

  void addExpr(MCInst &Inst, const MCExpr *Expr) const {
    assert(Expr && "Expr shouldn't be null!");
    int64_t Imm = 0;
    RISCVMCExpr::VariantKind VK = RISCVMCExpr::VK_RISCV_None;
    bool IsConstant = evaluateConstantImm(Expr, Imm, VK);

    if (IsConstant)
                                 ::createImm(Imm));
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

Show All 31 Lines

```
                                 createImm(Imm));
  }

  void addCSRSystemRegisterOperands(MCInst &Inst, unsigned N) const {
    assert(N == 1 && "Invalid number of operands!");
    Inst.addOperand(MCOperand::createImm(SysReg.Encoding));
  }

  void addVTypeIOperands(MCInst &Inst, unsigned N) const {
    assert(N == 1 && "Invalid number of operands!");
    Inst.addOperand(MCOperand::createImm(VType.Encoding));
  }

  // Returns the rounding mode represented by this RISCVOperand. Should only
  // be called after checking isFRMArg.
  RISCVFPRndMode::RoundingMode getRoundingMode() const {
    // isFRMArg has validated the operand, meaning this cast is safe.
    auto SE = cast<MCSymbolRefExpr>(getImm());
    RISCVFPRndMode::RoundingMode FRM =
        RISCVFPRndMode::stringToRoundingMode(SE->getSymbol().getName());
    assert(FRM != RISCVFPRndMode::Invalid && "Invalid rounding mode");
```

▲ Show 20 Lines • Show All 61 Lines • ▼ Show 20 Line(s)

```

  auto Result =
    MatchInstructionImpl(Operands, Inst, ErrorInfo, MissingFeatures,
                         MatchingInlineAsm);
  switch (Result) {
  default:
    break;
  case Match_Success:
    if (validateInstruction(Inst, Operands))
      return true;
    return processInstruction(Inst, IDLoc, Operands, Out);
  case Match_MissingFeature: {
    assert(MissingFeatures.any() && "Unknown missing features!");
    bool FirstFeature = true;
    std::string Msg = "instruction requires the following:";
    for (unsigned i = 0, e = MissingFeatures.size(); i != e; ++i) {
      if (MissingFeatures[i]) {
        Msg += FirstFeature ? " " : ", ";
```

▲ Show 20 Lines • Show All 156 Lines • ▼ Show 20 Line(s)

```
  case Match_InvalidCallSymbol: {
    SMLoc ErrorLoc = ((RISCVOperand &)*Operands[ErrorInfo]).getStartLoc();
    return Error(ErrorLoc, "operand must be a bare symbol name");
  }
  case Match_InvalidTPRelAddSymbol: {
    SMLoc ErrorLoc = ((RISCVOperand &)*Operands[ErrorInfo]).getStartLoc();
    return Error(ErrorLoc, "operand must be a symbol with %tprel_add modifier");
  }
  case Match_InvalidVTypeI: {
```

⏮ **evandro** ✔ **Done** ↶ ✖

Please, rename to `Match_InvalidVTypeI` .

```
    SMLoc ErrorLoc = ((RISCVOperand &)*Operands[ErrorInfo]).getStartLoc();
be e[8|16|32|64|128|256|512|1024],m[1|2|4|8]");
er: {
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

evandro ✔ Done

Please, rename to `Match_InvalidVMaskRegister`.

```
    SMLoc ErrorLoc = ((RISCVOperand &)*Operands[ErrorInfo]).getStartLoc();
    return Error(ErrorLoc, "operand must be v0.t");
  }
  }

  llvm_unreachable("Unknown match type detected!");
}

// Attempts to match Name as a register (either using the default name or
// alternative ABI names), setting RegNo to the matching register. Upon
// failure, returns true and sets RegNo to 0. If IsRV32E then registers
```

▲ Show 20 Lines • Show All 333 Lines • ▼ Show 20 Line(s)

```
  // is an identifier and is followed by a comma.
  if (getLexer().is(AsmToken::Identifier) &&
      getLexer().peekTok().is(AsmToken::Comma))
    return MatchOperand_NoMatch;

  return parseImmediate(Operands);
}

OperandMatchResultTy RISCVAsmParser::parseVTypeI(OperandVector &Operands) {
  SMLoc S = getLoc();
```

evandro ✔ Done

Please, rename to `parseVTypeI()`.

```
  if (getLexer().getKind() != AsmToken::Identifier)
    return MatchOperand_NoMatch;

  // Parse "e8,m1"
  StringRef Name = getLexer().getTok().getIdentifier();
  if (!Name.consume_front("e"))
    return MatchOperand_NoMatch;
  APInt Sew(16, Name, 10);
  if (Sew != 8 && Sew != 16 && Sew != 32 && Sew != 64 && Sew != 128 &&
      Sew != 256 && Sew != 512 && Sew != 1024)
    return MatchOperand_NoMatch;
  getLexer().Lex();

  if (getLexer().getKind() == AsmToken::EndOfStatement) {
    Operands.push_back(
        RISCVOperand::createVType(Sew, APInt(16, 1), S, isRV64()));

    return MatchOperand_Success;
  }

  if (!getLexer().is(AsmToken::Comma))
    return MatchOperand_NoMatch;
  getLexer().Lex();
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
tIdentifier();
;
  APInt Lmul(16, Name, 10);
  if (Lmul != 1 && Lmul != 2 && Lmul != 4 && Lmul != 8)
    return MatchOperand_NoMatch;
  getLexer().Lex();

  if (getLexer().getKind() != AsmToken::EndOfStatement)
    return MatchOperand_NoMatch;

  Operands.push_back(RISCVOperand::createVType(Sew, Lmul, S, isRV64()));

  return MatchOperand_Success;
}

OperandMatchResultTy RISCVAsmParser::parseMaskReg(OperandVector &Operands) {
  switch (getLexer().getKind()) {
  default:
```

evandro ✓ Done

Please, rename to `parseMaskReg()` .

```
    return MatchOperand_NoMatch;
  case AsmToken::Identifier:
    StringRef Name = getLexer().getTok().getIdentifier();
    if (!Name.consume_back(".t")) {
      Error(getLoc(), "expected '.t' suffix");
      return MatchOperand_ParseFail;
    }
    Register RegNo;
    matchRegisterNameHelper(isRV32E(), RegNo, Name);

    if (RegNo == RISCV::NoRegister)
```

rogfer01 Not Done

Shouldn't we check that `RegNo == RISCV::V0` here? I tried with

```
vadd.vv  v3, v4, v5, v0.t
vadd.vv  v3, v4, v5, v4.t
```

and I got

```
vadd.vv  v3, v4, v5, v0.t        # encoding: [0xd7,0x81,0x42,0x00]
vadd.vv  v3, v4, v5, v4.t        # encoding: [0xd7,0x81,0x42,0x00]
```

This doesn't seem right, does it?

```
      return MatchOperand_NoMatch;
    if (RegNo != RISCV::V0)
      return MatchOperand_NoMatch;
    SMLoc S = getLoc();
    SMLoc E = SMLoc::getFromPointer(S.getPointer() - 1);
    getLexer().Lex();
    Operands.push_back(RISCVOperand::createReg(RegNo, S, E, isRV64()));
  }
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
OperandMatchResultTy
```

```
1393 1590 RISCVAsmParser::parseMemOpBaseReg(OperandVector &Operands) {
1394 1591   if (getLexer().isNot(AsmToken::LParen)) {
1395 1592     Error(getLoc(), "expected '('");
1396 1593     return MatchOperand_ParseFail;
1397 1594   }
1398 1595
1399 1596   getParser().Lex(); // Eat '('
```

Show All 28 Lines

```
1428 1625   // operand that is silently dropped.
1429 1626   //
1430 1627   // Instead, we use this custom parser. This will: allow (and discard) an
1431 1628   // offset if it is zero; require (and discard) parentheses; and add only the
1432 1629   // parsed register operand to `Operands`.
1433 1630   //
1434 1631   // These operands are printed with RISCVInstPrinter::printAtomicMemOp, which
1435 1632   // will only print the register surrounded by parentheses (which GNU as also
1436 1633   // uses as its canonical representation for these operands).
```

rogfer01 Not Done

Could this encoding logic be in `RISCVOperand::createVType` itself?

You would have to pass `sew` and `lmul` without doing `logBase2()` here and the call in line `1395` would set `lmul = 1`.

```
1437 1634   std::unique_ptr<RISCVOperand> OptionalImmOp;
1438 1635
1439 1636   if (getLexer().isNot(AsmToken::LParen)) {
1440 1637     // Parse an Integer token. We do not accept arbitrary constant expressions
1441 1638     // in the offset field (because they may include parens, which complicates
1442 1639     // parsing a lot).
```

fpallares Not Done

There is an issue here: we are accepting an instruction like `vadd.vv v1, v2, v3, v0` (without `.t` suffix). This happens because `RISCVAsmParser::parseOperand` falls back to parse a register if a specific parsing like this function returns `MatchOperand_NoMatch`.

I've been looking into how to fix that and I believe we can diagnose an error here doing something like this:

```
if (!Name.consume_back(".t")) {
  Error(getLoc(), "expected '.t' suffix");
  return MatchOperand_ParseFail;
}
```

In summary:

- `v0.t` is accepted by the current parser, this is already OK.
- things like `v0.` or `v0.x` will be rejected by the proposed change.
- things like `v4.t` are already rejected by the `PredicateMethod` of `VRMaskAsmOperand`. I'd suggest adding a `DiagnosticType` in `VRMaskAsmOperand` in tablegen (`RISCVInstrInfoV.td`) [illegible]se diagnostic.

HsiangKai Author ✓ Done

It makes sense. I will modify the patch according to your comments. Thanks a lot.

```
    int64_t ImmVal;
    SMLoc ImmStart = getLoc();
    if (getParser().parseIntToken(ImmVal,
                                  "expected '(' or optional integer offset"))
      return MatchOperand_ParseFail;

    // Create a RISCVOperand for checking later (so the error messages are
    // nicer), but we don't add it to Operands.
```

▲ Show 20 Lines • Show All 588 Lines • ▼ Show 20 Line(s)

```
    SMLoc ErrorLoc = ((RISCVOperand &)*Operands[3]).getStartLoc();
    return Error(ErrorLoc, "the second input operand must be tp/x4 when using "
                           "%tprel_add modifier");
  }

  return false;
}

std::unique_ptr<RISCVOperand> RISCVAsmParser::defaultMaskRegOp() const {
```

rogfer01 Not Done

You plan to add tests for the erros diagnosed in this function as well, right?

HsiangKai Author ✓ Done

Yes, I will add more tests for these invalid combinations later.

```
  return RISCVOperand::createReg(RISCV::NoRegister, llvm::SMLoc(),
                                 llvm::SMLoc(), isRV64());
}

bool RISCVAsmParser::validateInstruction(MCInst &Inst,
                                         OperandVector &Operands) {
  const MCInstrDesc &MCID = MII.get(Inst.getOpcode());
```

fpallares Not Done

`Operands[1]` here might be an out-of-bounds access for some instructions with less than 2 operators (e.g. `RISCV::JALR`).

HsiangKai Author ✓ Done

Operands[0] in OperandVector will be operator. Operands[1] will be the first operand.
If there is an instruction with no operand, it might be an out-of-bounds access.

I will change the default location to point to operator. Thanks.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
onstraintOffset) & RISCV::ConstraintMask;
onstraint)
```

```
  unsigned DestReg = Inst.getOperand(0).getReg();
  // Operands[1] will be the first operand, DestReg.
  SMLoc Loc = Operands[1]->getStartLoc();
  if ((TargetFlags == RISCV::WidenV) || (TargetFlags == RISCV::WidenW) ||
      (TargetFlags == RISCV::SlideUp) || (TargetFlags == RISCV::Vrgather) ||
      (TargetFlags == RISCV::Vcompress)) {
    if (TargetFlags != RISCV::WidenW) {
      unsigned Src2Reg = Inst.getOperand(1).getReg();
      if (DestReg == Src2Reg)
        return Error(Loc, "The destination vector register group cannot overlap"
                          " the source vector register group.");
    }
    if (Inst.getOperand(2).isReg()) {
      unsigned Src1Reg = Inst.getOperand(2).getReg();
      if (DestReg == Src1Reg)
        return Error(Loc, "The destination vector register group cannot overlap"
                          " the source vector register group.");
    }
    if (Inst.getNumOperands() == 4) {
      unsigned MaskReg = Inst.getOperand(3).getReg();

      if (DestReg == MaskReg)
        return Error(Loc, "The destination vector register group cannot overlap"
                          " the mask register.");
    }
  } else if (TargetFlags == RISCV::Narrow) {
    unsigned Src2Reg = Inst.getOperand(1).getReg();
    if (DestReg == Src2Reg)
      return Error(Loc, "The destination vector register group cannot overlap"
                        " the source vector register group.");
  } else if (TargetFlags == RISCV::WidenCvt || TargetFlags == RISCV::Iota) {
    unsigned Src2Reg = Inst.getOperand(1).getReg();
    if (DestReg == Src2Reg)
      return Error(Loc, "The destination vector register group cannot overlap"
                        " the source vector register group.");
    if (Inst.getNumOperands() == 3) {
      unsigned MaskReg = Inst.getOperand(2).getReg();

      if (DestReg == MaskReg)
        return Error(Loc, "The destination vector register group cannot overlap"
                          " the mask register.");
    }
  }
  return false;
```

⏮ **rogfer01** **Not Done** ↩ ✖

There is a macro `LLVM_FALLTHROUGH` for these cases. You can use it here and in the later occurrences of a similar comment.

```
}

bool RISCVAsmParser::processInstruction(MCInst &Inst, SMLoc IDLoc,
                                        OperandVector &Operands,
                                        MCStreamer &Out) {
  Inst.setLoc(IDLoc);
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
    break;
```

▲ Show 20 Lines • Show All 92 Lines • Show Last 20 Lines

llvm/lib/Target/RISCV/Disassembler/RISCVDisassembler.cpp

☰ View Options

Show All 12 Lines

```
#include "MCTargetDesc/RISCVMCTargetDesc.h"
#include "TargetInfo/RISCVTargetInfo.h"
#include "Utils/RISCVBaseInfo.h"
#include "llvm/CodeGen/Register.h"
#include "llvm/MC/MCContext.h"
#include "llvm/MC/MCDisassembler/MCDisassembler.h"
#include "llvm/MC/MCFixedLenDisassembler.h"
#include "llvm/MC/MCInst.h"
#include "llvm/MC/MCInstrInfo.h"
#include "llvm/MC/MCRegisterInfo.h"
#include "llvm/MC/MCSubtargetInfo.h"
#include "llvm/Support/Endian.h"
#include "llvm/Support/TargetRegistry.h"

using namespace llvm;

#define DEBUG_TYPE "riscv-disassembler"

typedef MCDisassembler::DecodeStatus DecodeStatus;

namespace {
class RISCVDisassembler : public MCDisassembler {
  std::unique_ptr<MCInstrInfo const> const MCII;

public:
  RISCVDisassembler(const MCSubtargetInfo &STI, MCContext &Ctx)
      : MCDisassembler(STI, Ctx) {}
  RISCVDisassembler(const MCSubtargetInfo &STI, MCContext &Ctx,
                    MCInstrInfo const *MCII)
      : MCDisassembler(STI, Ctx), MCII(MCII) {}

  DecodeStatus getInstruction(MCInst &Instr, uint64_t &Size,
                              ArrayRef<uint8_t> Bytes, uint64_t Address,
                              raw_ostream &CStream) const override;
};
} // end anonymous namespace

static MCDisassembler *createRISCVDisassembler(const Target &T,
                                               const MCSubtargetInfo &STI,
                                               MCContext &Ctx) {
  return new RISCVDisassembler(STI, Ctx);
  return new RISCVDisassembler(STI, Ctx, T.createMCInstrInfo());
}

extern "C" LLVM_EXTERNAL_VISIBILITY void LLVMInitializeRISCVDisassembler() {
  // Register the disassembler for each target.
  TargetRegistry::RegisterMCDisassembler(getTheRISCV32Target(),
                                         createRISCVDisassembler);
  TargetRegistry::RegisterMCDisassembler(getTheRISCV64Target(),
                                         createRISCVDisassembler);
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

es • Show All 86 Lines • ▼ Show 20 Line(s)

```
  Register Reg = RISCV::X8 + RegNo;
  Inst.addOperand(MCOperand::createReg(Reg));
  return MCDisassembler::Success;
}

static DecodeStatus DecodeVRRegisterClass(MCInst &Inst, uint64_t RegNo,
```

fpallares Not Done

Following @evandro 's naming suggestions, this should probably be renamed to `DecodeVRegisterClass` .

HsiangKai Author ✓ Done

The function name will be expected by TableGen as `"Decode" + Record->getName().str() + "RegisterClass"` . It seems related to the name of register class. Currently, the name of RegisterClass for V-extension registers is "VR".

fpallares ✓ Done

I see, sorry for the noise.

```
                                          uint64_t Address,
                                          const void *Decoder) {
  if (RegNo >= 32)
    return MCDisassembler::Fail;

  Register Reg = RISCV::V0 + RegNo;
  Inst.addOperand(MCOperand::createReg(Reg));
  return MCDisassembler::Success;
}

static DecodeStatus decodeVMaskReg(MCInst &Inst, uint64_t RegNo,
```

evandro ✓ Done

Please, rename to `decodeVMaskReg()` .

```
                                   uint64_t Address, const void *Decoder) {
  Register Reg = RISCV::NoRegister;
  switch (RegNo) {
  default:
    return MCDisassembler::Fail;
  case 0:
    Reg = RISCV::V0;
    break;
  case 1:
    break;
  }
  Inst.addOperand(MCOperand::createReg(Reg));
  return MCDisassembler::Success;
}
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
nstructions *SP compressed instructions. The SP
oded in the instruction.
Inst, int64_t Address, const void *Decoder) {
  if (Inst.getOpcode() == RISCV::C_LWSP || Inst.getOpcode() == RISCV::C_SWSP ||
      Inst.getOpcode() == RISCV::C_LDSP || Inst.getOpcode() == RISCV::C_SDSP ||
      Inst.getOpcode() == RISCV::C_FLWSP ||
      Inst.getOpcode() == RISCV::C_FSWSP ||
      Inst.getOpcode() == RISCV::C_FLDSP ||
```

▲ Show 20 Lines • Show All 214 Lines • Show Last 20 Lines

llvm/lib/Target/RISCV/MCTargetDesc/RISCVInstPrinter.h

View Options

Show All 35 Lines

```
  void printCSRSystemRegister(const MCInst *MI, unsigned OpNo,
                              const MCSubtargetInfo &STI, raw_ostream &O);
  void printFenceArg(const MCInst *MI, unsigned OpNo,
                     const MCSubtargetInfo &STI, raw_ostream &O);
  void printFRMArg(const MCInst *MI, unsigned OpNo, const MCSubtargetInfo &STI,
                   raw_ostream &O);
  void printAtomicMemOp(const MCInst *MI, unsigned OpNo,
                        const MCSubtargetInfo &STI, raw_ostream &O);
  void printVTypeI(const MCInst *MI, unsigned OpNo, const MCSubtargetInfo &STI,
                   raw_ostream &O);
  void printVMaskReg(const MCInst *MI, unsigned OpNo,
```

evandro ✔ Done

Please, rename to `printVMaskReg()` .

```
                     const MCSubtargetInfo &STI, raw_ostream &O);

  // Autogenerated by tblgen.
  void printInstruction(const MCInst *MI, uint64_t Address,
                        const MCSubtargetInfo &STI, raw_ostream &O);
  bool printAliasInstr(const MCInst *MI, uint64_t Address,
                       const MCSubtargetInfo &STI, raw_ostream &O);
  void printCustomAliasOperand(const MCInst *MI, uint64_t Address,
                               unsigned OpIdx, unsigned PrintMethodIdx,
                               const MCSubtargetInfo &STI, raw_ostream &O);
  static const char *getRegisterName(unsigned RegNo);
  static const char *getRegisterName(unsigned RegNo, unsigned AltIdx);
};
} // namespace llvm

#endif
```

llvm/lib/Target/RISCV/MCTargetDesc/RISCVInstPrinter.cpp

View Options

Show First 20 Lines • Show All 144 Lines • ▼ Show 20 Line(s)

```

  assert(MO.isReg() && "printAtomicMemOp can only print register operands");
  O << "(";
  printRegName(O, MO.getReg());
  O << ")";
  return;
}

                                   peI(const MCInst *MI, unsigned OpNo,
                                       const MCSubtargetInfo &STI, raw_ostream &O) {
                                   (OpNo).getImm();
  unsigned Sew = (Imm >> 2) & 0x7;
  unsigned Lmul = Imm & 0x3;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```

  Lmul = 0x1 << Lmul;
  Sew = 0x1 << (Sew + 3);
  O << "e" << Sew << ",m" << Lmul;
}

void RISCVInstPrinter::printVMaskReg(const MCInst *MI, unsigned OpNo,
```

⏮ **evandro** ✔ Done ↩ ✖

Please, rename to `printVMaskReg()` .

```
                                     const MCSubtargetInfo &STI,
                                     raw_ostream &O) {
  const MCOperand &MO = MI->getOperand(OpNo);

  assert(MO.isReg() && "printVMaskReg can only print register operands");
  if (MO.getReg() == RISCV::NoRegister)
    return;
  O << ", ";
  printRegName(O, MO.getReg());
  O << ".t";
}

const char *RISCVInstPrinter::getRegisterName(unsigned RegNo) {
  return getRegisterName(RegNo, ArchRegNames ? RISCV::NoRegAltName
                                             : RISCV::ABIRegAltName);
}
```

llvm/lib/Target/RISCV/MCTargetDesc/RISCVMCCodeEmitter.cpp

≡ View Options

Show First 20 Lines • Show All 74 Lines • ▼ Show 20 Line(s)

```

  unsigned getImmOpValueAsr1(const MCInst &MI, unsigned OpNo,
                             SmallVectorImpl<MCFixup> &Fixups,
                             const MCSubtargetInfo &STI) const;

  unsigned getImmOpValue(const MCInst &MI, unsigned OpNo,
                         SmallVectorImpl<MCFixup> &Fixups,
                         const MCSubtargetInfo &STI) const;

  unsigned getVMaskReg(const MCInst &MI, unsigned OpNo,
```

⏮ **evandro** ✔ Done ↩ ✖

Please, rename to `getVMaskReg()` .

```
                       SmallVectorImpl<MCFixup> &Fixups,
                       const MCSubtargetInfo &STI) const;
};
} // end anonymous namespace

MCCodeEmitter *llvm::createRISCVMCCodeEmitter(const MCInstrInfo &MCII,
                                              const MCRegisterInfo &MRI,
                                              MCContext &Ctx) {
                                            (Ctx, MCII);
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

• Show All 278 Lines • ▼ Show 20 Line(s)

```
    MCFixup::create(0, Dummy, MCFixupKind(RISCV::fixup_riscv_relax),
                    MI.getLoc()));
    ++MCNumFixups;
  }

  return 0;
}

unsigned RISCVMCCodeEmitter::getVMaskReg(const MCInst &MI, unsigned OpNo,
                                         SmallVectorImpl<MCFixup> &Fixups,
                                         const MCSubtargetInfo &STI) const {
  MCOperand MO = MI.getOperand(OpNo);
  assert(MO.isReg() && "Expected a register.");

  switch (MO.getReg()) {
  default:
    llvm_unreachable("Invalid mask register.");
  case RISCV::V0:
    return 0;
  case RISCV::NoRegister:
    return 1;
  }
}

#include "RISCVGenMCCodeEmitter.inc"
```

llvm/lib/Target/RISCV/RISCV.td

☰ View Options

Show First 20 Lines • Show All 141 Lines • ▼ Show 20 Line(s)

```

def FeatureRVCHints
    : SubtargetFeature<"rvc-hints", "EnableRVCHintInstrs", "true",
                       "Enable RVC Hint Instructions.">;
def HasRVCHints : Predicate<"Subtarget->enableRVCHintInstrs()">,
                            AssemblerPredicate<(all_of FeatureRVCHints),
                            "RVC Hint Instructions">;

def FeatureStdExtV
    : SubtargetFeature<"experimental-v", "HasStdExtV", "true",
                       "'V' (Vector Instructions)">;
```

evandro Not Done

Should it also imply `FeatureStdExtD` ?

HsiangKai Author ✓ Done

I didn't find that V will imply D extension. Could you indicate the description in spec about this?

evandro Not Done

Indeed it doesn't:

> *…clude floating-point, then a `fcsr` register is also added to hold*
> *…m CSRs as explained below.*

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

Which means that the instructions that take FP scalars should also depend on `hasStdExt{F|D}`. Likewise when the scalar is a 64 bit integer, `hasRV64`.

rogfer01 Not Done

I don't think we have to do anything special with 64 bit: section 11.1 states what is the behaviour when `XLEN` is different than `SEW`.

evandro Not Done

Good point. 👍

HsiangKai Author ✓ Done

Yes, it is a problem about accessing fcsr if F extension is turned off. However, I found that there is vcsr in the latest version.

https://github.com/riscv/riscv-v-spec/commit/b25b643b9c29b4fde570293c64ca682a99a09f2a

So, I think we could keep V and F as separate unrelated extensions.

I will add predicates for vector floating point operations.

```
def HasStdExtV : Predicate<"Subtarget->hasStdExtV()">,
                           AssemblerPredicate<(all_of FeatureStdExtV),
```

fpallares ✓ Done

Since D69899 was commited, we should add the feature name in the `AssemblerPredicate` parameter and update the missing feature error messages in the tests. Like this:

```
def HasStdExtV : Predicate<"Subtarget->hasStdExtV()">,
                           AssemblerPredicate<"FeatureStdExtV",
                           "'V' (Vector Instructions)">;
```

```
                           "'V' (Vector Instructions)">;

def Feature64Bit
    : SubtargetFeature<"64bit", "HasRV64", "true", "Implements RV64">;
def IsRV64 : Predicate<"Subtarget->is64Bit()">,
                       AssemblerPredicate<(all_of Feature64Bit),
                       "RV64I Base Instruction Set">;
def IsRV32 : Predicate<"!Subtarget->is64Bit()">,
                       AssemblerPredicate<(all_of (not Feature64Bit)),
                       "RV32I Base Instruction Set">;
```

▲ Show 20 Lines • Show All 78 Lines • Show Last 20 Lines

llvm/lib/Target/RISCV/RISCVInstrFormats.td

☰ View Options

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

es • Show All 43 Lines • ▼ Show 20 Line(s)

```
at<11>;
at<12>;
def InstFormatCS     : InstFormat<13>;
def InstFormatCA     : InstFormat<14>;
def InstFormatCB     : InstFormat<15>;
def InstFormatCJ     : InstFormat<16>;
def InstFormatOther  : InstFormat<17>;

class RISCVVConstraint<bits<4> val> {
  bits<4> Value = val;
}
def NoConstraint : RISCVVConstraint<0>;
def WidenV       : RISCVVConstraint<1>;
def WidenW       : RISCVVConstraint<2>;
def WidenCvt     : RISCVVConstraint<3>;
def QuadWiden    : RISCVVConstraint<4>;
def Narrow       : RISCVVConstraint<5>;
def Iota         : RISCVVConstraint<6>;
def SlideUp      : RISCVVConstraint<7>;
def Vrgather     : RISCVVConstraint<8>;
def Vcompress    : RISCVVConstraint<9>;

// The following opcode names match those given in Table 19.1 in the
// RISC-V User-level ISA specification ("RISC-V base opcode map").
class RISCVOpcode<bits<7> val> {
  bits<7> Value = val;
}
def OPC_LOAD      : RISCVOpcode<0b0000011>;
def OPC_LOAD_FP   : RISCVOpcode<0b0000111>;
def OPC_MISC_MEM  : RISCVOpcode<0b0001111>;
def OPC_OP_IMM    : RISCVOpcode<0b0010011>;
def OPC_AUIPC     : RISCVOpcode<0b0010111>;
def OPC_OP_IMM_32 : RISCVOpcode<0b0011011>;
def OPC_STORE     : RISCVOpcode<0b0100011>;
def OPC_STORE_FP  : RISCVOpcode<0b0100111>;
def OPC_AMO       : RISCVOpcode<0b0101111>;
def OPC_OP        : RISCVOpcode<0b0110011>;
def OPC_LUI       : RISCVOpcode<0b0110111>;
def OPC_OP_32     : RISCVOpcode<0b0111011>;
def OPC_MADD      : RISCVOpcode<0b1000011>;
def OPC_MSUB      : RISCVOpcode<0b1000111>;
def OPC_NMSUB     : RISCVOpcode<0b1001011>;
def OPC_NMADD     : RISCVOpcode<0b1001111>;
def OPC_OP_FP     : RISCVOpcode<0b1010011>;
def OPC_OP_V      : RISCVOpcode<0b1010111>;
def OPC_BRANCH    : RISCVOpcode<0b1100011>;
def OPC_JALR      : RISCVOpcode<0b1100111>;
def OPC_JAL       : RISCVOpcode<0b1101111>;
def OPC_SYSTEM    : RISCVOpcode<0b1110011>;

class RVInst<dag outs, dag ins, string opcodestr, string argstr,
             list<dag> pattern, InstFormat format>
    : Instruction {
```

Show All 12 Lines

```
  let Namespace = "RISCV";

  dag OutOperandList = outs;
  dag InOperandList = ins;
  let AsmString = opcodestr # "\t" # argstr;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
ue;
```

```
  // Defaults
  RISCVVConstraint RVVConstraint = NoConstraint;
  let TSFlags{8-5} = RVVConstraint.Value;
}

// Pseudo instructions
class Pseudo<dag outs, dag ins, list<dag> pattern, string opcodestr = "", string argstr = "">
    : RVInst<outs, ins, opcodestr, argstr, pattern, InstFormatPseudo>,
      Sched<[]> {
  let isPseudo = 1;
  let isCodeGenOnly = 1;
```

▲ Show 20 Lines • Show All 206 Lines • Show Last 20 Lines

llvm/lib/Target/RISCV/RISCVInstrFormatsV.td

☰ View Options

This file was added.

```
//===-- RISCVInstrFormatsV.td - RISCV V Instruction Formats --*- tablegen -*-=//
//
// Part of the LLVM Project, under the Apache License v2.0 with LLVM Exceptions.
// See https://llvm.org/LICENSE.txt for license information.
// SPDX-License-Identifier: Apache-2.0 WITH LLVM-exception
//
//===----------------------------------------------------------------------===//
//
//  This file describes the RISC-V V extension instruction formats.
//
//===----------------------------------------------------------------------===//

class RISCVVFormat<bits<3> val> {
  bits<3> Value = val;
}
def OPIVV : RISCVVFormat<0b000>;
def OPFVV : RISCVVFormat<0b001>;
def OPMVV : RISCVVFormat<0b010>;
def OPIVI : RISCVVFormat<0b011>;
def OPIVX : RISCVVFormat<0b100>;
def OPFVF : RISCVVFormat<0b101>;
def OPMVX : RISCVVFormat<0b110>;

class RISCVMOP<bits<3> val> {
  bits<3> Value = val;
}
def MOPLDUnitStrideU : RISCVMOP<0b000>;
def MOPLDStridedU    : RISCVMOP<0b010>;
def MOPLDIndexedU    : RISCVMOP<0b011>;
def MOPLDUnitStrideS : RISCVMOP<0b100>;
def MOPLDStridedS    : RISCVMOP<0b110>;
def MOPLDIndexedS    : RISCVMOP<0b111>;

def MOPSTUnitStride  : RISCVMOP<0b000>;
def MOPSTStrided     : RISCVMOP<0b010>;
def MOPSTIndexedOrder: RISCVMOP<0b011>;
def MOPSTIndexedUnOrd: RISCVMOP<0b111>;

                                {

def LUMOPUnitStride  : RISCVLSUMOP<0b00000>;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
def LUMOPUnitStrideWholeReg : RISCVLSUMOP<0b01000>;
def LUMOPUnitStrideFF: RISCVLSUMOP<0b10000>;
def SUMOPUnitStride  : RISCVLSUMOP<0b00000>;
def SUMOPUnitStrideWholeReg : RISCVLSUMOP<0b01000>;

class RISCVWidth<bits<3> val> {
  bits<3> Value = val;
}
def LSWidthVByte : RISCVWidth<0b000>;
def LSWidthVHalf : RISCVWidth<0b101>;
def LSWidthVWord : RISCVWidth<0b110>;
def LSWidthVSEW  : RISCVWidth<0b111>;

class RVInstSetVLi<dag outs, dag ins, string opcodestr, string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatI> {
  bits<5> rs1;
  bits<5> rd;
  bits<11> vtypei;

  let Inst{31} = 0;
  let Inst{30-20} = vtypei;
  let Inst{19-15} = rs1;
  let Inst{14-12} = 0b111;
  let Inst{11-7} = rd;
  let Opcode = OPC_OP_V.Value;

  let Defs = [VTYPE, VL];
}

class RVInstSetVL<dag outs, dag ins, string opcodestr, string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> rs2;
  bits<5> rs1;
  bits<5> rd;

  let Inst{31} = 1;
  let Inst{30-25} = 0b000000;
  let Inst{24-20} = rs2;
  let Inst{19-15} = rs1;
  let Inst{14-12} = 0b111;
  let Inst{11-7} = rd;
  let Opcode = OPC_OP_V.Value;

  let Defs = [VTYPE, VL];
}

class RVInstVV<bits<6> funct6, RISCVVFormat opv, dag outs, dag ins,
               string opcodestr, string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> vs2;
  bits<5> vs1;
  bits<5> vd;
  bit vm;

  let Inst{31-26} = funct6;
  let Inst{25} = vm;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
  let Opcode = OPC_OP_V.Value;

  let Uses = [VTYPE, VL];
}

class RVInstVX<bits<6> funct6, RISCVVFormat opv, dag outs, dag ins,
                string opcodestr, string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> vs2;
  bits<5> rs1;
  bits<5> vd;
  bit vm;

  let Inst{31-26} = funct6;
  let Inst{25} = vm;
  let Inst{24-20} = vs2;
  let Inst{19-15} = rs1;
  let Inst{14-12} = opv.Value;
  let Inst{11-7} = vd;
  let Opcode = OPC_OP_V.Value;

  let Uses = [VTYPE, VL];
}

class RVInstV2<bits<6> funct6, bits<5> vs2, RISCVVFormat opv, dag outs, dag ins,
                string opcodestr, string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> rs1;
  bits<5> vd;
  bit vm;

  let Inst{31-26} = funct6;
  let Inst{25} = vm;
  let Inst{24-20} = vs2;
  let Inst{19-15} = rs1;
  let Inst{14-12} = opv.Value;
  let Inst{11-7} = vd;
  let Opcode = OPC_OP_V.Value;

  let Uses = [VTYPE, VL];
}

class RVInstIVI<bits<6> funct6, dag outs, dag ins, string opcodestr,
                string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> vs2;
  bits<5> imm;
  bits<5> vd;
  bit vm;

  let Inst{31-26} = funct6;
  let Inst{25} = vm;
  let Inst{24-20} = vs2;
  let Inst{19-15} = imm;
  let Inst{14-12} = 0b011;
  let Inst{11-7} = vd;
  let Opcode = OPC_OP_V.Value;
```

```
class RVInstV<bits<6> funct6, bits<5> vs1, RISCVVFormat opv, dag outs,
              dag ins, string opcodestr, string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> vs2;
  bits<5> vd;
  bit vm;

  let Inst{31-26} = funct6;
  let Inst{25} = vm;
  let Inst{24-20} = vs2;
  let Inst{19-15} = vs1;
  let Inst{14-12} = opv.Value;
  let Inst{11-7} = vd;
  let Opcode = OPC_OP_V.Value;

  let Uses = [VTYPE, VL];
}

class RVInstVLU<bits<3> nf, RISCVMOP mop, RISCVLSUMOP lumop,
                RISCVWidth width, dag outs, dag ins, string opcodestr,
                string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> rs1;
  bits<5> vd;
  bit vm;

  let Inst{31-29} = nf;
  let Inst{28-26} = mop.Value;
  let Inst{25} = vm;
  let Inst{24-20} = lumop.Value;
  let Inst{19-15} = rs1;
  let Inst{14-12} = width.Value;
  let Inst{11-7} = vd;
  let Opcode = OPC_LOAD_FP.Value;

  let Uses = [VTYPE, VL];
}

class RVInstVLS<bits<3> nf, RISCVMOP mop, RISCVWidth width,
                dag outs, dag ins, string opcodestr, string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> rs2;
  bits<5> rs1;
  bits<5> vd;
  bit vm;

  let Inst{31-29} = nf;
  let Inst{28-26} = mop.Value;
  let Inst{25} = vm;
  let Inst{24-20} = rs2;
  let Inst{19-15} = rs1;
  let Inst{14-12} = width.Value;
  let Inst{11-7} = vd;
  let Opcode = OPC_LOAD_FP.Value;

  let Uses = [VTYPE, VL];
}

CVMOP mop, RISCVWidth width,
ns, string opcodestr, string argstr>
tr, argstr, [], InstFormatR> {
  bits<5> vs2;
  bits<5> rs1;
  bits<5> vd;
  bit vm;

  let Inst{31-29} = nf;
  let Inst{28-26} = mop.Value;
  let Inst{25} = vm;
  let Inst{24-20} = vs2;
  let Inst{19-15} = rs1;
  let Inst{14-12} = width.Value;
  let Inst{11-7} = vd;
  let Opcode = OPC_LOAD_FP.Value;

  let Uses = [VTYPE, VL];
}

class RVInstVSU<bits<3> nf, RISCVMOP mop, RISCVLSUMOP sumop,
                RISCVWidth width, dag outs, dag ins, string opcodestr,
                string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> rs1;
  bits<5> vs3;
  bit vm;

  let Inst{31-29} = nf;
  let Inst{28-26} = mop.Value;
  let Inst{25} = vm;
  let Inst{24-20} = sumop.Value;
  let Inst{19-15} = rs1;
  let Inst{14-12} = width.Value;
  let Inst{11-7} = vs3;
  let Opcode = OPC_STORE_FP.Value;

  let Uses = [VTYPE, VL];
}

class RVInstVSS<bits<3> nf, RISCVMOP mop, RISCVWidth width,
                dag outs, dag ins, string opcodestr, string argstr>
    : RVInst<outs, ins, opcodestr, argstr, [], InstFormatR> {
  bits<5> rs2;
  bits<5> rs1;
  bits<5> vs3;
  bit vm;

  let Inst{31-29} = nf;
  let Inst{28-26} = mop.Value;
  let Inst{25} = vm;
  let Inst{24-20} = rs2;
  let Inst{19-15} = rs1;
  let Inst{14-12} = width.Value;
  let Inst{11-7} = vs3;
  let Opcode = OPC_STORE_FP.Value;

  let Uses = [VTYPE, VL];
}

class RVInstVSX<bits<3> nf, RISCVMOP mop, RISCVWidth width,
                              ns, string opcodestr, string argstr>
                              tr, argstr, [], InstFormatR> {

  bits<5> rs1;
  bits<5> vs3;
  bit vm;

  let Inst{31-29} = nf;
  let Inst{28-26} = mop.Value;
  let Inst{25} = vm;
  let Inst{24-20} = vs2;
  let Inst{19-15} = rs1;
  let Inst{14-12} = width.Value;
  let Inst{11-7} = vs3;
  let Opcode = OPC_STORE_FP.Value;

  let Uses = [VTYPE, VL];
}
```

llvm/lib/Target/RISCV/RISCVInstrInfo.h

☰ View Options

Show First 20 Lines • Show All 127 Lines • ▼ Show 20 Line(s)

```
  // Insert a call to an outlined function into a given basic block.
  virtual MachineBasicBlock::iterator
  insertOutlinedCall(Module &M, MachineBasicBlock &MBB,
                     MachineBasicBlock::iterator &It, MachineFunction &MF,
                     const outliner::Candidate &C) const override;
protected:
  const RISCVSubtarget &STI;
};
}

namespace RISCV {
// Match with the definitions in RISCVInstrFormatsV.td
enum RVVConstraintType {
  NoConstraint = 0,
  WidenV = 1,
  WidenW = 2,
  WidenCvt = 3,
  QuadWiden = 4,
  Narrow = 5,
  Iota = 6,
  SlideUp = 7,
  Vrgather = 8,
  Vcompress = 9,

  ConstraintOffset = 5,
  ConstraintMask = 0b1111
};
} // end namespace RISCV

} // end namespace llvm
#endif
```

llvm/lib/Target/RISCV/RISCVInstrInfo.td

☰ View Options

Show First 20 Lines • Show All 1159 Lines • ▼ Show 20 Line(s)

```
//===----------------------------------------------------------------------===//
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
include "RISCVInstrInfoF.td"
include "RISCVInstrInfoD.td"
include "RISCVInstrInfoC.td"
include "RISCVInstrInfoB.td"
include "RISCVInstrInfoV.td"
```

llvm/lib/Target/RISCV/RISCVInstrInfoV.td

≡ View Options

This file was added.

```
//===-- RISCVInstrInfoV.td - RISC-V 'V' instructions -------*- tablegen -*-===//
//
// Part of the LLVM Project, under the Apache License v2.0 with LLVM Exceptions.
// See https://llvm.org/LICENSE.txt for license information.
// SPDX-License-Identifier: Apache-2.0 WITH LLVM-exception
//
//===----------------------------------------------------------------------===//
///
/// This file describes the RISC-V instructions from the standard 'V',
/// Vector instruction set extension.
///
//===----------------------------------------------------------------------===//

include "RISCVInstrFormatsV.td"

//===----------------------------------------------------------------------===//
// Operand and SDNode transformation definitions.
//===----------------------------------------------------------------------===//

def VTypeIAsmOperand : AsmOperandClass {
  let Name = "VTypeI";
  let ParserMethod = "parseVTypeI";
  let DiagnosticType = "InvalidVTypeI";
}

def VTypeIOp : Operand<XLenVT> {
  let ParserMatchClass = VTypeIAsmOperand;
  let PrintMethod = "printVTypeI";
  let DecoderMethod = "decodeUImmOperand<11>";
}

def VRegAsmOperand : AsmOperandClass {
  let Name = "RVVRegOpOperand";
```

⏮ **evandro** ✔ Done ↩ ✖

Please, rename to `VRegAsmOperand` .

```
  let RenderMethod = "addRegOperands";
  let PredicateMethod = "isReg";
  let ParserMethod = "parseRegister";
}

def VRegOp : RegisterOperand<VR> {
  let ParserMatchClass = VRegAsmOperand;
```

⏮ **evandro** ✔ Done ↩ ✖

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

fpallares Not Done

Whitespace missing before the colon.

```
  let PrintMethod = "printOperand";
}

def VMaskAsmOperand : AsmOperandClass {
  let Name = "RVVMaskRegOpOperand";
```

evandro ✓ Done

Please, rename to `VMaskAsmOperand` .

```
  let RenderMethod = "addRegOperands";
  let PredicateMethod = "isV0Reg";
  let ParserMethod = "parseMaskReg";
  let IsOptional = 1;
  let DefaultMethod = "defaultMaskRegOp";
  let DiagnosticType = "InvalidVMaskRegister";
}

def VMaskOp : RegisterOperand<VMV0> {
  let ParserMatchClass = VMaskAsmOperand;
```

evandro ✓ Done

Please, rename to `VMaskOp` .

fpallares Not Done

Ditto.

```
  let PrintMethod = "printVMaskReg";
  let EncoderMethod = "getVMaskReg";
  let DecoderMethod = "decodeVMaskReg";
}

def simm5 : Operand<XLenVT>, ImmLeaf<XLenVT, [{return isInt<5>(Imm);}]> {
  let ParserMatchClass = SImmAsmOperand<5>;
  let EncoderMethod = "getImmOpValue";
  let DecoderMethod = "decodeSImmOperand<5>";
  let MCOperandPredicate = [{
    int64_t Imm;
    if (MCOp.evaluateAsConstantImm(Imm))
      return isInt<5>(Imm);
    return MCOp.isBareSymbolRef();
  }];
}

//===----------------------------------------------------------------------===//
// Instruction class templates
//===----------------------------------------------------------------------===//

= 1, mayStore = 0 in {
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

evandro Not Done

No need to set fields to the default `0` .

HsiangKai Author ✓ Done

There is no default value for hasSideEffects, mayLoad, and mayStore.

evandro Not Done

Right. It would be cleaner if they were moved to the respective base instruction class then. Just a suggestion.

```
class VUnitStrideLoad<RISCVMOP mop, RISCVLSUMOP lumop, RISCVWidth width,
                      string opcodestr>
    : RVInstVLU<0b000, mop, lumop, width, (outs VRegOp:$vd),
                (ins GPR:$rs1, VMaskOp:$vm), opcodestr, "$vd, (${rs1})$vm">;

// load vd, (rs1), rs2, vm
class VStridedLoad<RISCVMOP mop, RISCVWidth width, string opcodestr>
    : RVInstVLS<0b000, mop, width, (outs VRegOp:$vd),
                (ins GPR:$rs1, GPR:$rs2, VMaskOp:$vm), opcodestr,
                "$vd, (${rs1}), $rs2$vm">;

// load vd, (rs1), vs2, vm
class VIndexedLoad<RISCVMOP mop, RISCVWidth width, string opcodestr>
    : RVInstVLX<0b000, mop, width, (outs VRegOp:$vd),
                (ins GPR:$rs1, VRegOp:$vs2, VMaskOp:$vm), opcodestr,
                "$vd, (${rs1}), $vs2$vm">;

// vl<nf>r.v vd, (rs1)
class VWholeLoad<bits<3> nf, string opcodestr>
    : RVInstVLU<nf, MOPLDUnitStrideU, LUMOPUnitStrideWholeReg,
                LSWidthVSEW, (outs VRegOp:$vd), (ins GPR:$rs1),
                opcodestr, "$vd, (${rs1})"> {
  let vm = 1;
  let Uses = [];
}
}

let hasSideEffects = 0, mayLoad = 0, mayStore = 1 in {
// store vd, vs3, (rs1), vm
```

fpallares Not Done

I don't see why setting `hasSideEffects` is necessary and setting `mayStore` is not enough, could you please elaborate a bit on this?

evandro Not Done

`hasSideEffects` is not necessary, since `mayStore` is indeed enough. The purpose of `hasSideEffects` is as a catch all when the rest of the instruction attributes miss some side effect.

✓ Done

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

There is no default value for hasSideEffects, mayLoad, and mayStore in `Instruction` class. So, I specify these three values for V instructions. It follows the style in RISCVInstrInfo.td. I agree that we could give these attributes some default values in some base class for V instructions. I will try to refactor it.

fpallares Not Done

I was only referring tot he fact that `hasSideEffects` here changed from `0` to `1` in a recent update of the patch. I don't really think a refactor is necessary here.

```
class VUnitStrideStore<RISCVMOP mop, RISCVLSUMOP sumop, RISCVWidth width,
                         string opcodestr>
    : RVInstVSU<0b000, mop, sumop, width, (outs),
                (ins VRegOp:$vs3, GPR:$rs1, VMaskOp:$vm), opcodestr,
                "$vs3, (${rs1})$vm">;

// store vd, vs3, (rs1), rs2, vm
class VStridedStore<RISCVMOP mop, RISCVWidth width, string opcodestr>
    : RVInstVSS<0b000, mop, width, (outs),
                (ins VRegOp:$vs3, GPR:$rs1, GPR:$rs2, VMaskOp:$vm),
                opcodestr, "$vs3, (${rs1}), $rs2$vm">;

// store vd, vs3, (rs1), vs2, vm
class VIndexedStore<RISCVMOP mop, RISCVWidth width, string opcodestr>
    : RVInstVSX<0b000, mop, width, (outs),
                (ins VRegOp:$vs3, GPR:$rs1, VRegOp:$vs2, VMaskOp:$vm),
                opcodestr, "$vs3, (${rs1}), $vs2$vm">;

// vs<nf>r.v vd, (rs1)
class VWholeStore<bits<3> nf, string opcodestr>
    : RVInstVSU<nf, MOPSTUnitStride, SUMOPUnitStrideWholeReg,
                LSWidthVSEW, (outs), (ins VRegOp:$vs3, GPR:$rs1),
                opcodestr, "$vs3, (${rs1})"> {
  let vm = 1;
  let Uses = [];
}
}

let hasSideEffects = 0, mayLoad = 0, mayStore = 0 in {
// op vd, vs2, vs1, vm
class VALUVV<bits<6> funct6, RISCVVFormat opv, string opcodestr>
    : RVInstVV<funct6, opv, (outs VRegOp:$vd),
                (ins VRegOp:$vs2, VRegOp:$vs1, VMaskOp:$vm),
                opcodestr, "$vd, $vs2, $vs1$vm">;

// op vd, vs2, vs1, v0 (without mask, use v0 as carry input)
class VALUmVV<bits<6> funct6, RISCVVFormat opv, string opcodestr>
    : RVInstVV<funct6, opv, (outs VRegOp:$vd),
                (ins VRegOp:$vs2, VRegOp:$vs1, VMV0:$v0),
                opcodestr, "$vd, $vs2, $vs1, v0"> {
  let vm = 0;
}

                                the order of vs1 and vs2)
                               ISCVVFormat opv, string opcodestr>
                               ts VRegOp:$vd),
                               1, VRegOp:$vs2, VMaskOp:$vm),
                opcodestr, "$vd, $vs1, $vs2$vm">;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```

// op vd, vs1, vs2
class VALUVVNoVm<bits<6> funct6, RISCVVFormat opv, string opcodestr>
    : RVInstVV<funct6, opv, (outs VRegOp:$vd),
               (ins VRegOp:$vs2, VRegOp:$vs1),
               opcodestr, "$vd, $vs2, $vs1"> {
  let vm = 1;
}

// op vd, vs2, rs1, vm
class VALUVX<bits<6> funct6, RISCVVFormat opv, string opcodestr>
    : RVInstVX<funct6, opv, (outs VRegOp:$vd),
                (ins VRegOp:$vs2, GPR:$rs1, VMaskOp:$vm),
                opcodestr, "$vd, $vs2, $rs1$vm">;

// op vd, vs2, rs1, v0 (without mask, use v0 as carry input)
class VALUmVX<bits<6> funct6, RISCVVFormat opv, string opcodestr>
    : RVInstVX<funct6, opv, (outs VRegOp:$vd),
                (ins VRegOp:$vs2, GPR:$rs1, VMV0:$v0),
                opcodestr, "$vd, $vs2, $rs1, v0"> {
  let vm = 0;
}

// op vd, rs1, vs2, vm (reverse the order of rs1 and vs2)
class VALUrVX<bits<6> funct6, RISCVVFormat opv, string opcodestr>
    : RVInstVX<funct6, opv, (outs VRegOp:$vd),
                (ins GPR:$rs1, VRegOp:$vs2, VMaskOp:$vm),
                opcodestr, "$vd, $rs1, $vs2$vm">;

// op vd, vs1, vs2
class VALUVXNoVm<bits<6> funct6, RISCVVFormat opv, string opcodestr>
    : RVInstVX<funct6, opv, (outs VRegOp:$vd),
               (ins VRegOp:$vs2, GPR:$rs1),
               opcodestr, "$vd, $vs2, $rs1"> {
  let vm = 1;
}

// op vd, vs2, imm, vm
class VALUVI<bits<6> funct6, string opcodestr, Operand optype = simm5>
    : RVInstIVI<funct6, (outs VRegOp:$vd),
                (ins VRegOp:$vs2, optype:$imm, VMaskOp:$vm),
                opcodestr, "$vd, $vs2, $imm$vm">;

// op vd, vs2, imm, v0 (without mask, use v0 as carry input)
class VALUmVI<bits<6> funct6, string opcodestr, Operand optype = simm5>
    : RVInstIVI<funct6, (outs VRegOp:$vd),
                (ins VRegOp:$vs2, optype:$imm, VMV0:$v0),
                opcodestr, "$vd, $vs2, $imm, v0"> {
  let vm = 0;
}

// op vd, vs2, imm, vm
class VALUVINoVm<bits<6> funct6, string opcodestr, Operand optype = simm5>
    : RVInstIVI<funct6, (outs VRegOp:$vd),
                (ins VRegOp:$vs2, optype:$imm),
                opcodestr, "$vd, $vs2, $imm"> {
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
class VALUVF<bits<6> funct6, RISCVVFormat opv, string opcodestr>
```

```
    : RVInstVX<funct6, opv, (outs VRegOp:$vd),
                (ins VRegOp:$vs2, FPR32:$rs1, VMaskOp:$vm),
                opcodestr, "$vd, $vs2, $rs1$vm">;

// op vd, rs1, vs2, vm (Float) (with mask, reverse the order of rs1 and vs2)
class VALUrVF<bits<6> funct6, RISCVVFormat opv, string opcodestr>
    : RVInstVX<funct6, opv, (outs VRegOp:$vd),
                (ins FPR32:$rs1, VRegOp:$vs2, VMaskOp:$vm),
                opcodestr, "$vd, $rs1, $vs2$vm">;

// op vd, vs2, vm (use vs1 as instruction encoding)
class VALUVs2<bits<6> funct6, bits<5> vs1, RISCVVFormat opv, string opcodestr>
    : RVInstV<funct6, vs1, opv, (outs VRegOp:$vd),
               (ins VRegOp:$vs2, VMaskOp:$vm),
               opcodestr, "$vd, $vs2$vm">;
}

//===----------------------------------------------------------------------===//
// Combination of instruction classes.
// Use these multiclasses to define instructions more easily.
//===----------------------------------------------------------------------===//
multiclass VALU_IV_V_X_I<string opcodestr, bits<6> funct6, Operand optype = simm5, string vw = "v"> {
  def V  : VALUVV<funct6, OPIVV, opcodestr # "." # vw # "v">;
  def X  : VALUVX<funct6, OPIVX, opcodestr # "." # vw # "x">;
  def I  : VALUVI<funct6, opcodestr # "." # vw # "i", optype>;
}

multiclass VALU_IV_V_X<string opcodestr, bits<6> funct6, string vw = "v"> {
  def V  : VALUVV<funct6, OPIVV, opcodestr # "." # vw # "v">;
  def X  : VALUVX<funct6, OPIVX, opcodestr # "." # vw # "x">;
}

multiclass VALUr_IV_V_X<string opcodestr, bits<6> funct6, string vw = "v"> {
  def V : VALUrVV<funct6, OPIVV, opcodestr # "." # vw # "v">;
  def X : VALUrVX<funct6, OPIVX, opcodestr # "." # vw # "x">;
}

multiclass VALU_IV_X_I<string opcodestr, bits<6> funct6, Operand optype = simm5, string vw = "v"> {
  def X  : VALUVX<funct6, OPIVX, opcodestr # "." # vw # "x">;
  def I  : VALUVI<funct6, opcodestr # "." # vw # "i", optype>;
}

multiclass VALU_IV_V<string opcodestr, bits<6> funct6> {
  def _VS  : VALUVV<funct6, OPIVV, opcodestr # ".vs">;
}

multiclass VALUr_IV_X<string opcodestr, bits<6> funct6, string vw = "v"> {
  def X : VALUrVX<funct6, OPIVX, opcodestr # "." # vw # "x">;
}

multiclass VALU_MV_V_X<string opcodestr, bits<6> funct6, string vw = "v"> {
  def V : VALUVV<funct6, OPMVV, opcodestr # "." # vw # "v">;
  def X  : VALUVX<funct6, OPMVX, opcodestr # "." # vw # "x">;
}

                                   odestr, bits<6> funct6> {
                                  V, opcodestr # ".vs">;
```

```
multiclass VALU_MV_Mask<string opcodestr, bits<6> funct6, string vm = "v"> {
  def M : VALUVVNoVm<funct6, OPMVV, opcodestr # "." # vm # "m">;
}

multiclass VALU_MV_X<string opcodestr, bits<6> funct6, string vw = "v"> {
  def X  : VALUVX<funct6, OPMVX, opcodestr # "." # vw # "x">;
}

multiclass VALUr_MV_V_X<string opcodestr, bits<6> funct6, string vw = "v"> {
  def V : VALUrVV<funct6, OPMVV, opcodestr # "." # vw # "v">;
  def X : VALUrVX<funct6, OPMVX, opcodestr # "." # vw # "x">;
}

multiclass VALUr_MV_X<string opcodestr, bits<6> funct6, string vw = "v"> {
  def X : VALUrVX<funct6, OPMVX, opcodestr # "." # vw # "x">;
}

multiclass VALU_MV_VS2<string opcodestr, bits<6> funct6, bits<5> vs1> {
  def "" : VALUVs2<funct6, vs1, OPMVV, opcodestr>;
}

multiclass VALUm_IV_V_X_I<string opcodestr, bits<6> funct6> {
  def VM : VALUmVV<funct6, OPIVV, opcodestr # ".vvm">;
  def XM : VALUmVX<funct6, OPIVX, opcodestr # ".vxm">;
  def IM : VALUmVI<funct6, opcodestr # ".vim">;
}

multiclass VALUm_IV_V_X<string opcodestr, bits<6> funct6> {
  def VM : VALUmVV<funct6, OPIVV, opcodestr # ".vvm">;
  def XM : VALUmVX<funct6, OPIVX, opcodestr # ".vxm">;
}

multiclass VALUNoVm_IV_V_X_I<string opcodestr, bits<6> funct6, Operand optype = simm5> {
  def V : VALUVVNoVm<funct6, OPIVV, opcodestr # ".vv">;
  def X : VALUVXNoVm<funct6, OPIVX, opcodestr # ".vx">;
  def I : VALUVINoVm<funct6, opcodestr # ".vi", optype>;
}

multiclass VALUNoVm_IV_V_X<string opcodestr, bits<6> funct6> {
  def V : VALUVVNoVm<funct6, OPIVV, opcodestr # ".vv">;
  def X : VALUVXNoVm<funct6, OPIVX, opcodestr # ".vx">;
}

multiclass VALU_FV_V_F<string opcodestr, bits<6> funct6, string vw = "v"> {
  def V : VALUVV<funct6, OPFVV, opcodestr # "." # vw # "v">;
  def F : VALUVF<funct6, OPFVF, opcodestr # "." # vw # "f">;
}

multiclass VALU_FV_F<string opcodestr, bits<6> funct6, string vw = "v"> {
  def F : VALUVF<funct6, OPFVF, opcodestr # "." # vw # "f">;
}

multiclass VALUr_FV_V_F<string opcodestr, bits<6> funct6, string vw = "v"> {
  def V : VALUrVV<funct6, OPFVV, opcodestr # "." # vw # "v">;
  def F : VALUrVF<funct6, OPFVF, opcodestr # "." # vw # "f">;
}

                                    odestr, bits<6> funct6> {
                                    V, opcodestr # ".vs">;

```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
multiclass VALU_FV_VS2<string opcodestr, bits<6> funct6, bits<5> vs1> {
  def "" : VALUVs2<funct6, vs1, OPFVV, opcodestr>;
}

//===----------------------------------------------------------------------===//
// Instructions
//===----------------------------------------------------------------------===//

let Predicates = [HasStdExtV] in {

let hasSideEffects = 1, mayLoad = 0, mayStore = 0 in {
def VSETVLI : RVInstSetVLi<(outs GPR:$rd), (ins GPR:$rs1, VTypeIOp:$vtypei),
                           "vsetvli", "$rd, $rs1, $vtypei">;

def VSETVL : RVInstSetVL<(outs GPR:$rd), (ins GPR:$rs1, GPR:$rs2),
                         "vsetvl", "$rd, $rs1, $rs2">;
}

// Vector Unit-Stride Instructions
def VLB_V : VUnitStrideLoad<MOPLDUnitStrideS, LUMOPUnitStride, LSWidthVByte, "vlb.v">;
def VLH_V : VUnitStrideLoad<MOPLDUnitStrideS, LUMOPUnitStride, LSWidthVHalf, "vlh.v">;
def VLW_V : VUnitStrideLoad<MOPLDUnitStrideS, LUMOPUnitStride, LSWidthVWord, "vlw.v">;

def VLBU_V : VUnitStrideLoad<MOPLDUnitStrideU, LUMOPUnitStride, LSWidthVByte, "vlbu.v">;
def VLHU_V : VUnitStrideLoad<MOPLDUnitStrideU, LUMOPUnitStride, LSWidthVHalf, "vlhu.v">;
def VLWU_V : VUnitStrideLoad<MOPLDUnitStrideU, LUMOPUnitStride, LSWidthVWord, "vlwu.v">;

def VLE_V : VUnitStrideLoad<MOPLDUnitStrideU, LUMOPUnitStride, LSWidthVSEW, "vle.v">;

def VLBFF_V : VUnitStrideLoad<MOPLDUnitStrideS, LUMOPUnitStrideFF, LSWidthVByte, "vlbff.v">;
def VLHFF_V : VUnitStrideLoad<MOPLDUnitStrideS, LUMOPUnitStrideFF, LSWidthVHalf, "vlhff.v">;
def VLWFF_V : VUnitStrideLoad<MOPLDUnitStrideS, LUMOPUnitStrideFF, LSWidthVWord, "vlwff.v">;

def VLBUFF_V : VUnitStrideLoad<MOPLDUnitStrideU, LUMOPUnitStrideFF, LSWidthVByte, "vlbuff.v">;
def VLHUFF_V : VUnitStrideLoad<MOPLDUnitStrideU, LUMOPUnitStrideFF, LSWidthVHalf, "vlhuff.v">;
def VLWUFF_V : VUnitStrideLoad<MOPLDUnitStrideU, LUMOPUnitStrideFF, LSWidthVWord, "vlwuff.v">;

def VLEFF_V : VUnitStrideLoad<MOPLDUnitStrideU, LUMOPUnitStrideFF, LSWidthVSEW, "vleff.v">;

def VSB_V : VUnitStrideStore<MOPSTUnitStride, SUMOPUnitStride, LSWidthVByte, "vsb.v">;
def VSH_V : VUnitStrideStore<MOPSTUnitStride, SUMOPUnitStride, LSWidthVHalf, "vsh.v">;
def VSW_V : VUnitStrideStore<MOPSTUnitStride, SUMOPUnitStride, LSWidthVWord, "vsw.v">;

def VSE_V : VUnitStrideStore<MOPSTUnitStride, SUMOPUnitStride, LSWidthVSEW, "vse.v">;

// Vector Strided Instructions
def VLSB_V : VStridedLoad<MOPLDStridedS, LSWidthVByte, "vlsb.v">;
def VLSH_V : VStridedLoad<MOPLDStridedS, LSWidthVHalf, "vlsh.v">;
def VLSW_V : VStridedLoad<MOPLDStridedS, LSWidthVWord, "vlsw.v">;

def VLSBU_V : VStridedLoad<MOPLDStridedU, LSWidthVByte, "vlsbu.v">;
def VLSHU_V : VStridedLoad<MOPLDStridedU, LSWidthVHalf, "vlshu.v">;
def VLSWU_V : VStridedLoad<MOPLDStridedU, LSWidthVWord, "vlswu.v">;

                                          StridedU, LSWidthVSEW, "vlse.v">;

                                          TStrided, LSWidthVByte, "vssb.v">;
def VSSH_V : VStridedStore<MOPSTStrided, LSWidthVHalf, "vssh.v">;
def VSSW_V : VStridedStore<MOPSTStrided, LSWidthVWord, "vssw.v">;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
def VSSE_V : VStridedStore<MOPSTStrided, LSWidthVSEW, "vsse.v">;

// Vector Indexed Instructions
def VLXB_V : VIndexedLoad<MOPLDIndexedS, LSWidthVByte, "vlxb.v">;
def VLXH_V : VIndexedLoad<MOPLDIndexedS, LSWidthVHalf, "vlxh.v">;
def VLXW_V : VIndexedLoad<MOPLDIndexedS, LSWidthVWord, "vlxw.v">;

def VLXBU_V : VIndexedLoad<MOPLDIndexedU, LSWidthVByte, "vlxbu.v">;
def VLXHU_V : VIndexedLoad<MOPLDIndexedU, LSWidthVHalf, "vlxhu.v">;
def VLXWU_V : VIndexedLoad<MOPLDIndexedU, LSWidthVWord, "vlxwu.v">;

def VLXE_V : VIndexedLoad<MOPLDIndexedU, LSWidthVSEW, "vlxe.v">;

def VSXB_V : VIndexedStore<MOPSTIndexedOrder, LSWidthVByte, "vsxb.v">;
def VSXH_V : VIndexedStore<MOPSTIndexedOrder, LSWidthVHalf, "vsxh.v">;
def VSXW_V : VIndexedStore<MOPSTIndexedOrder, LSWidthVWord, "vsxw.v">;
def VSXE_V : VIndexedStore<MOPSTIndexedOrder, LSWidthVSEW, "vsxe.v">;

def VSUXB_V : VIndexedStore<MOPSTIndexedUnOrd, LSWidthVByte, "vsuxb.v">;
def VSUXH_V : VIndexedStore<MOPSTIndexedUnOrd, LSWidthVHalf, "vsuxh.v">;
def VSUXW_V : VIndexedStore<MOPSTIndexedUnOrd, LSWidthVWord, "vsuxw.v">;
def VSUXE_V : VIndexedStore<MOPSTIndexedUnOrd, LSWidthVSEW, "vsuxe.v">;

def VL1R_V : VWholeLoad<0, "vl1r.v">;
def VS1R_V : VWholeStore<0, "vs1r.v">;
```

fpallares Not Done

As of the stable v0.8 release, the whole register vector loads and stores have been restricted to a single register only. You probably just forgot to change this `foreach` since `vl1r.v` and `vs1r.v` are the only instructions being tested.

After this change you may want to revisit whether the `VWholeLoad` and `VWholeStore` multiclasses are still needed.

```

// Vector Single-Width Integer Add and Subtract
defm VADD_V : VALU_IV_V_X_I<"vadd", 0b000000>;
defm VSUB_V : VALU_IV_V_X<"vsub", 0b000010>;
defm VRSUB_V : VALU_IV_X_I<"vrsub", 0b000011>;

```

fpallares Not Done

Whitespace missing before the colon.

```
// Vector Widening Integer Add/Subtract
// Refer to 11.2 Widening Vector Arithmetic Instructions
// The destination vector register group cannot overlap a source vector
// register group of a different element width (including the mask register
// if masked), otherwise an illegal instruction exception is raised.
let Constraints = "@earlyclobber $vd" in {
let RVVConstraint = WidenV in {
defm VWADDU_V : VALU_MV_V_X<"vwaddu", 0b110000>;
defm VWSUBU_V : VALU_MV_V_X<"vwsubu", 0b110010>;
dd", 0b110001>;
ub", 0b110011>;
```

fpallares

⏮ Not Done ↩ ✖

I'd say those instructions need the `@earlyclobber $vd` restriction too. Not because of the first source operand (since it has the same width as the destination), but because of the second (in the case of `VWOP_WV`), or the mask (for `VWOP_WX`).

Quoting section 11.2. Widening Vector Arithmetic Instructions:

> *The destination vector register group cannot overlap a source vector register group of a different element width (including the mask register if masked), otherwise an illegal instruction exception is raised.*

This has the downside that the `earlyclobber` constraint is too coarse and will impose unnecessary restrictions by not allowing the destination to overlap with the first (wide) operand, even when the instruction is used unmasked. Maybe we can use the `earlyclobber` for now, and find a better solution for this in a later patch.

```
// Set earlyclobber for following instructions for second and mask operands.
// This has the downside that the earlyclobber constraint is too coarse and
// will impose unnecessary restrictions by not allowing the destination to
// overlap with the first (wide) operand.
let RVVConstraint = WidenW in {
defm VWADDU_W : VALU_MV_V_X<"vwaddu", 0b110100, "w">;
defm VWSUBU_W : VALU_MV_V_X<"vwsubu", 0b110110, "w">;
defm VWADD_W : VALU_MV_V_X<"vwadd", 0b110101, "w">;
defm VWSUB_W : VALU_MV_V_X<"vwsub", 0b110111, "w">;
}
}

// Vector Integer Add-with-Carry / Subtract-with-Borrow Instructions
defm VADC_V : VALUm_IV_V_X_I<"vadc", 0b010000>;
defm VMADC_V : VALUm_IV_V_X_I<"vmadc", 0b010001>;
defm VMADC_V : VALUNoVm_IV_V_X_I<"vmadc", 0b010001>;
defm VSBC_V : VALUm_IV_V_X<"vsbc", 0b010010>;
defm VMSBC_V : VALUm_IV_V_X<"vmsbc", 0b010011>;
defm VMSBC_V : VALUNoVm_IV_V_X<"vmsbc", 0b010011>;

// Vector Bitwise Logical Instructions
defm VAND_V : VALU_IV_V_X_I<"vand", 0b001001>;
```

⏮ **fpallares** Not Done ↩ ✖

In section 11.3. Narrowing Vector Arithmetic Instructions we find the following paragraph:

> *The destination vector register group cannot overlap the first source vector register group (specified by vs2). The destination vector register group cannot overlap the mask register if used, unless LMUL=1. If either constraint is violated, an illegal instruction exception is raised.*

From my understanding, this applies to `VNSRL` and `VNSRA` . In that case we need an `@earlyclobber` constraint here.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

✔ Done ↩ ✖

In current implementation for MC layer, there is no LMUL information in these instructions. So, if LMUL=1, there is no such restriction for them. I have a downstream patch to extend these instruction definitions for LMUL > 1. Maybe we could keep this MC implementation for LMUL=1 and extend it later. I will add comments for it.

rogfer01 Not Done

Perhaps the wording in the spec is unclear here?

Looks like the following sentence does apply regardless of LMUL.

> > *The destination vector register group cannot overlap the first source vector register group (specified by vs2).*

While the sentence that follows it is clear that applies only to LMUL>1

> > *The destination vector register group cannot overlap the mask register if used, unless LMUL=1*

You also said that

> *I have a downstream patch to extend these instruction definitions for LMUL > 1*

I'm curious here: did you use a Pseudo for that or somehow avoided the issue that they'd be encoded in the same way?

Thanks!

HsiangKai Author ✓ Done

I misunderstood the statements. Sorry for that. I will add earlyclobber for narrowing instructions. Thanks a lot.

About instruction definitions for LMUL > 1, I enable 'isCodeGenOnly' for these instructions to avoid encoding checking.

```
defm VOR_V : VALU_IV_V_X_I<"vor", 0b001010>;
defm VXOR_V : VALU_IV_V_X_I<"vxor", 0b001011>;

// Vector Single-Width Bit Shift Instructions
defm VSLL_V : VALU_IV_V_X_I<"vsll", 0b100101, uimm5>;
defm VSRL_V : VALU_IV_V_X_I<"vsrl", 0b101000, uimm5>;
defm VSRA_V : VALU_IV_V_X_I<"vsra", 0b101001, uimm5>;

// Vector Narrowing Integer Right Shift Instructions
// Refer to 11.3. Narrowing Vector Arithmetic Instructions
// The destination vector register group cannot overlap the first source
                                 fied by vs2). The destination vector register
                                 k register if used, unless LMUL=1.
                                 r $vd", RVVConstraint = Narrow in {
                                 nsrl", 0b101100, uimm5, "w">;
defm VNSRA_W : VALU_IV_V_X_I<"vnsra", 0b101101, uimm5, "w">;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
}

// Vector Integer Comparison Instructions
defm VMSEQ_V : VALU_IV_V_X_I<"vmseq", 0b011000>;
```

⏮ **fpallares** Not Done ↩ ✖

Whitespace missing before the colon.

```
defm VMSNE_V : VALU_IV_V_X_I<"vmsne", 0b011001>;
defm VMSLTU_V : VALU_IV_V_X<"vmsltu", 0b011010>;
defm VMSLT_V : VALU_IV_V_X<"vmslt", 0b011011>;
defm VMSLEU_V : VALU_IV_V_X_I<"vmsleu", 0b011100>;
defm VMSLE_V : VALU_IV_V_X_I<"vmsle", 0b011101>;
defm VMSGTU_V : VALU_IV_X_I<"vmsgtu", 0b011110>;
defm VMSGT_V : VALU_IV_X_I<"vmsgt", 0b011111>;

// Vector Integer Min/Max Instructions
defm VMINU_V : VALU_IV_V_X<"vminu", 0b000100>;
defm VMIN_V : VALU_IV_V_X<"vmin", 0b000101>;
defm VMAXU_V : VALU_IV_V_X<"vmaxu", 0b000110>;
defm VMAX_V : VALU_IV_V_X<"vmax", 0b000111>;

// Vector Single-Width Integer Multiply Instructions
defm VMUL_V : VALU_MV_V_X<"vmul", 0b100101>;
defm VMULH_V : VALU_MV_V_X<"vmulh", 0b100111>;
defm VMULHU_V : VALU_MV_V_X<"vmulhu", 0b100100>;
defm VMULHSU_V : VALU_MV_V_X<"vmulhsu", 0b100110>;

// Vector Integer Divide Instructions
defm VDIVU_V : VALU_MV_V_X<"vdivu", 0b100000>;
defm VDIV_V : VALU_MV_V_X<"vdiv", 0b100001>;
defm VREMU_V : VALU_MV_V_X<"vremu", 0b100010>;
defm VREM_V : VALU_MV_V_X<"vrem", 0b100011>;

// Vector Widening Integer Multiply Instructions
let Constraints = "@earlyclobber $vd", RVVConstraint = WidenV in {
defm VWMUL_V : VALU_MV_V_X<"vwmul", 0b111011>;
defm VWMULU_V : VALU_MV_V_X<"vwmulu", 0b111000>;
defm VWMULSU_V : VALU_MV_V_X<"vwmulsu", 0b111010>;
```

⏮ **fpallares** ✔ Done ↩ ✖

I wonder if this should be `VMERGE_V`, so that the defined records are `VMERGE_VVM`, `VMERGE_VXM`, `VMERGE_VIM`.

Currently these records are defined `VMERGE_VM`, `VMERGE_XM` and `VMERGE_IM`.

```
}

// Vector Single-Width Integer Multiply-Add Instructions
defm VMACC_V : VALUr_MV_V_X<"vmacc", 0b101101>;
defm VNMSAC_V : VALUr_MV_V_X<"vnmsac", 0b101111>;
defm VMADD_V : VALUr_MV_V_X<"vmadd", 0b101001>;
defm VNMSUB_V : VALUr_MV_V_X<"vnmsub", 0b101011>;

                  iply-Add Instructions
                  r $vd", RVVConstraint = WidenV in {
                  vwmaccu", 0b111100>;
defm VWMACC_V : VALUr_MV_V_X<"vwmacc", 0b111101>;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
defm VWMACCSU_V : VALUr_MV_V_X<"vwmaccsu", 0b111111>;
defm VWMACCUS_V : VALUr_MV_X<"vwmaccus", 0b111110>;
}

// Vector Integer Merge Instructions
defm VMERGE_V : VALUm_IV_V_X_I<"vmerge", 0b010111>;

// Vector Integer Move Instructions
let hasSideEffects = 0, mayLoad = 0, mayStore = 0 in {
// op vd, vs1
def VMV_V_V : RVInstVV<0b010111, OPIVV, (outs VRegOp:$vd),
                       (ins VRegOp:$vs1), "vmv.v.v", "$vd, $vs1"> {
```

evandro Not Done

Shouldn't these have `vxsat` in `Defs` ?

HsiangKai Author ✓ Done

It should have no impact on CodeGen. It seems no instructions to use vxsat.

evandro Not Done

For the sake of completeness.

```
  let vs2 = 0;
  let vm = 1;
}
// op vd, rs1
def VMV_V_X : RVInstVX<0b010111, OPIVX, (outs VRegOp:$vd),
                       (ins GPR:$rs1), "vmv.v.x", "$vd, $rs1"> {
```

evandro Not Done

Shouldn't these have `vxrm` in `Uses` ?

```
  let vs2 = 0;
  let vm = 1;
}
// op vd, imm
def VMV_V_I : RVInstIVI<0b010111, (outs VRegOp:$vd),
                       (ins simm5:$imm), "vmv.v.i", "$vd, $imm"> {
```

evandro Not Done

Shouldn't this hace `vxrm` in `Uses` and `vxsat` in `Defs` ?

```
  let vs2 = 0;
  let vm = 1;
}
```

evandro Not Done

ension?

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
// Vector Fixed-Point Arithmetic Instructions
defm VSADDU_V : VALU_IV_V_X_I<"vsaddu", 0b100000>;
defm VSADD_V : VALU_IV_V_X_I<"vsadd", 0b100001>;
defm VSSUBU_V : VALU_IV_V_X<"vssubu", 0b100010>;
```

evandro Not Done

Shouldn't this hace `vxrm` in `Uses`?

```
defm VSSUB_V : VALU_IV_V_X<"vssub", 0b100011>;
```

fpallares Not Done

I'd say overlap restrictions on narrowing instructions apply to `VNCLIP` and `VNCLIPU` too.

```

// Vector Single-Width Averaging Add and Subtract
defm VAADDU_V : VALU_MV_V_X<"vaaddu", 0b001000>;
```

evandro Not Done

Shouldn't this hace `vxrm` in `Uses` and `vxsat` in `Defs`?

```
defm VAADD_V : VALU_MV_V_X<"vaadd", 0b001001>;
defm VASUBU_V : VALU_MV_V_X<"vasubu", 0b001010>;
defm VASUB_V : VALU_MV_V_X<"vasub", 0b001011>;

// Vector Single-Width Fractional Multiply with Rounding and Saturation
defm VSMUL_V : VALU_IV_V_X<"vsmul", 0b100111>;

// Vector Single-Width Scaling Shift Instructions
defm VSSRL_V : VALU_IV_V_X_I<"vssrl", 0b101010, uimm5>;
defm VSSRA_V : VALU_IV_V_X_I<"vssra", 0b101011, uimm5>;

```

fpallares Not Done

As discussed in another inline comment above (for `VWADD_W` and such), there is an overlap restriction for these instructions too.

```
// Vector Narrowing Fixed-Point Clip Instructions
let Constraints = "@earlyclobber $vd", RVVConstraint = Narrow in {
defm VNCLIPU_W : VALU_IV_V_X_I<"vnclipu", 0b101110, uimm5, "w">;
defm VNCLIP_W : VALU_IV_V_X_I<"vnclip", 0b101111, uimm5, "w">;
}

// Vector Single-Width Integer Reduction Instructions
defm VREDSUM : VALU_MV_V<"vredsum", 0b000000>;
defm VREDMAXU : VALU_MV_V<"vredmaxu", 0b000110>;
defm VREDMAX : VALU_MV_V<"vredmax", 0b000111>;
defm VREDMINU : VALU_MV_V<"vredminu", 0b000100>;
defm VREDMIN : VALU_MV_V<"vredmin", 0b000101>;
defm VREDAND : VALU_MV_V<"vredand", 0b000001>;
```

fpallares Not Done

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

lon.

```
defm VREDOR : VALU_MV_V<"vredor", 0b000010>;
defm VREDXOR : VALU_MV_V<"vredxor", 0b000011>;

// Vector Widening Integer Reduction Instructions
let Constraints = "@earlyclobber $vd" in {
// Set earlyclobber for following instructions for second and mask operands.
// This has the downside that the earlyclobber constraint is too coarse and
// will impose unnecessary restrictions by not allowing the destination to
// overlap with the first (wide) operand.
defm VWREDSUMU : VALU_IV_V<"vwredsumu", 0b110000>;
defm VWREDSUM : VALU_IV_V<"vwredsum", 0b110001>;
}

// Vector Mask-Register Logical Instructions
defm VMAND_M : VALU_MV_Mask<"vmand", 0b011001, "m">;
defm VMNAND_M : VALU_MV_Mask<"vmnand", 0b011101, "m">;
defm VMANDNOT_M : VALU_MV_Mask<"vmandnot", 0b011000, "m">;
defm VMXOR_M : VALU_MV_Mask<"vmxor", 0b011011, "m">;
defm VMOR_M : VALU_MV_Mask<"vmor", 0b011010, "m">;
defm VMNOR_M : VALU_MV_Mask<"vmnor", 0b011110, "m">;
defm VMORNOT_M : VALU_MV_Mask<"vmornot", 0b011100, "m">;
defm VMXNOR_M : VALU_MV_Mask<"vmxnor", 0b011111, "m">;

let hasSideEffects = 0, mayLoad = 0, mayStore = 0 in {
// Vector mask population count vpopc
def VPOPC_M : RVInstV<0b010000, 0b10000, OPMVV, (outs GPR:$vd),
                        (ins VRegOp:$vs2, VMaskOp:$vm),
                        "vpopc.m", "$vd, $vs2$vm">;

// vfirst find-first-set mask bit
def VFIRST_M : RVInstV<0b010000, 0b10001, OPMVV, (outs GPR:$vd),
                        (ins VRegOp:$vs2, VMaskOp:$vm),
                        "vfirst.m", "$vd, $vs2$vm">;
}

// vmsbf.m set-before-first mask bit
defm VMSBF_M : VALU_MV_VS2<"vmsbf.m", 0b010100, 0b00001>;

// vmsif.m set-including-first mask bit
defm VMSIF_M : VALU_MV_VS2<"vmsif.m", 0b010100, 0b00011>;

// vmsof.m set-only-first mask bit
defm VMSOF_M : VALU_MV_VS2<"vmsof.m", 0b010100, 0b00010>;

// Vector Iota Instruction
let Constraints = "@earlyclobber $vd", RVVConstraint = Iota in {
defm VIOTA_M : VALU_MV_VS2<"viota.m", 0b010100, 0b10000>;
}

// Vector Element Index Instruction
let hasSideEffects = 0, mayLoad = 0, mayStore = 0 in {
def VID_V : RVInstV<0b010100, 0b10001, OPMVV, (outs VRegOp:$vd),
                      (ins VMaskOp:$vm), "vid.v", "$vd$vm"> {
  let vs2 = 0;
}

// Integer Scalar Move Instructions
[illegible]
                              0b00000, OPMVV, (outs GPR:$vd),
                             Op:$vs2), "vmv.x.s", "$vd, $vs2">;
                             , 0b00000, OPMVX, (outs VRegOp:$vd),
                      (ins GPR:$rs1), "vmv.s.x", "$vd, $rs1">;
```

```

}

foreach nf = [1, 2, 4, 8] in {
  def VMV#nf#R_V  : RVInstV<0b100111, !add(nf, -1), OPIVI, (outs VRegOp:$vd),
                            (ins VRegOp:$vs2), "vmv" # nf # "r.v",
                            "$vd, $vs2"> {
    let Uses = [];
    let vm = 1;
  }
}
}

// Vector Slide Instructions
let Constraints = "@earlyclobber $vd", RVVConstraint = SlideUp in {
```

⏮ **fpallares** Not Done ↩ ✖

I believe we are also missing the `@earlyclobber $vd` overlap constraints for these narrowing conversions.

From section 14.17. Narrowing Floating-Point/Integer Type-Convert Instructions:

> *These instructions have the same constraints on vector register overlap as other narrowing instructions (see Narrowing Vector Arithmetic Instructions).*

```
defm VSLIDEUP_V : VALU_IV_X_I<"vslideup", 0b001110, uimm5>;
}
defm VSLIDEDOWN_V : VALU_IV_X_I<"vslidedown", 0b001111, uimm5>;

let Constraints = "@earlyclobber $vd", RVVConstraint = SlideUp in {
defm VSLIDE1UP_V : VALU_MV_X<"vslide1up", 0b001110>;
}
defm VSLIDE1DOWN_V : VALU_MV_X<"vslide1down", 0b001111>;

// Vector Register Gather Instruction
let Constraints = "@earlyclobber $vd", RVVConstraint = Vrgather in {
defm VRGATHER_V : VALU_IV_V_X_I<"vrgather", 0b001100, uimm5>;
}

```

⏮ **fpallares** Not Done ↩ ✖

If I'm not mistaken, those instructions should have the `@earlyclobber $vd` constraint too to avoid overlaps.

```
// Vector Compress Instruction
let Constraints = "@earlyclobber $vd", RVVConstraint = Vcompress in {
defm VCOMPRESS_V : VALU_MV_Mask<"vcompress", 0b010111>;
}
} // Predicates = [HasStdExtV]

let Predicates = [HasStdExtF, HasStdExtV] in {
// Vector Single-Width Floating-Point Add/Subtract Instructions
dd", 0b000000>;
ub", 0b000010>;
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

Not Done ↩ ✖

Ditto.

```
defm VFRSUB_V : VALU_FV_F<"vfrsub", 0b100111>;

// Vector Widening Floating-Point Add/Subtract Instructions
let Constraints = "@earlyclobber $vd" in {
let RVVConstraint = WidenV in {
defm VFWADD_V : VALU_FV_V_F<"vfwadd", 0b110000>;
defm VFWSUB_V : VALU_FV_V_F<"vfwsub", 0b110010>;
}
// Set earlyclobber for following instructions for second and mask operands.
// This has the downside that the earlyclobber constraint is too coarse and
// will impose unnecessary restrictions by not allowing the destination to
// overlap with the first (wide) operand.
let RVVConstraint = WidenW in {
defm VFWADD_W : VALU_FV_V_F<"vfwadd", 0b110100, "w">;
defm VFWSUB_W : VALU_FV_V_F<"vfwsub", 0b110110, "w">;
}
}

// Vector Single-Width Floating-Point Multiply/Divide Instructions
defm VFMUL_V : VALU_FV_V_F<"vfmul", 0b100100>;
defm VFDIV_V : VALU_FV_V_F<"vfdiv", 0b100000>;
defm VFRDIV_V : VALU_FV_F<"vfrdiv", 0b100001>;

// Vector Widening Floating-Point Multiply
let Constraints = "@earlyclobber $vd", RVVConstraint = WidenV in {
defm VFWMUL_V : VALU_FV_V_F<"vfwmul", 0b111000>;
}

// Vector Single-Width Floating-Point Fused Multiply-Add Instructions
defm VFMACC_V : VALUr_FV_V_F<"vfmacc", 0b101100>;
defm VFNMACC_V : VALUr_FV_V_F<"vfnmacc", 0b101101>;
defm VFMSAC_V : VALUr_FV_V_F<"vfmsac", 0b101110>;
defm VFNMSAC_V : VALUr_FV_V_F<"vfnmsac", 0b101111>;
defm VFMADD_V : VALUr_FV_V_F<"vfmadd", 0b101000>;
defm VFNMADD_V : VALUr_FV_V_F<"vfnmadd", 0b101001>;
defm VFMSUB_V : VALUr_FV_V_F<"vfmsub", 0b101010>;
defm VFNMSUB_V : VALUr_FV_V_F<"vfnmsub", 0b101011>;

// Vector Widening Floating-Point Fused Multiply-Add Instructions
let Constraints = "@earlyclobber $vd", RVVConstraint = WidenV in {
defm VFWMACC_V : VALUr_FV_V_F<"vfwmacc", 0b111100>;
defm VFWNMACC_V : VALUr_FV_V_F<"vfwnmacc", 0b111101>;
defm VFWMSAC_V : VALUr_FV_V_F<"vfwmsac", 0b111110>;
defm VFWNMSAC_V : VALUr_FV_V_F<"vfwnmsac", 0b111111>;
}

// Vector Floating-Point Square-Root Instruction
defm VFSQRT_V : VALU_FV_VS2<"vfsqrt.v", 0b100011, 0b00000>;

// Vector Floating-Point MIN/MAX Instructions
defm VFMIN_V : VALU_FV_V_F<"vfmin", 0b000100>;
defm VFMAX_V : VALU_FV_V_F<"vfmax", 0b000110>;

// Vector Floating-Point Sign-Injection Instructions
sgnj", 0b001000>;
fsgnjn", 0b001001>;
fsgnjx", 0b001010>;

// Vector Floating-Point Compare Instructions
```

```
defm VMFEQ_V : VALU_FV_V_F<"vmfeq", 0b011000>;
defm VMFNE_V : VALU_FV_V_F<"vmfne", 0b011100>;
defm VMFLT_V : VALU_FV_V_F<"vmflt", 0b011011>;
defm VMFLE_V : VALU_FV_V_F<"vmfle", 0b011001>;
defm VMFGT_V : VALU_FV_F<"vmfgt", 0b011101>;
defm VMFGE_V : VALU_FV_F<"vmfge", 0b011111>;

// Vector Floating-Point Classify Instruction
defm VFCLASS_V : VALU_FV_VS2<"vfclass.v", 0b100011, 0b10000>;

let hasSideEffects = 0, mayLoad = 0, mayStore = 0 in {
// Vector Floating-Point Merge Instruction
def VFMERGE_VFM : RVInstVX<0b010111, OPFVF, (outs VRegOp:$vd),
                           (ins VRegOp:$vs2, FPR32:$rs1, VMV0:$v0),
                           "vfmerge.vfm", "$vd, $vs2, $rs1, v0"> {
  let vm = 0;
}

// Vector Floating-Point Move Instruction
def VFMV_V_F : RVInstVX<0b010111, OPFVF, (outs VRegOp:$vd),
                       (ins FPR32:$rs1), "vfmv.v.f", "$vd, $rs1"> {
  let vs2 = 0;
  let vm = 1;
}
}

// Single-Width Floating-Point/Integer Type-Convert Instructions
defm VFCVT_XU_F_V : VALU_FV_VS2<"vfcvt.xu.f.v", 0b100010, 0b00000>;
defm VFCVT_X_F_V : VALU_FV_VS2<"vfcvt.x.f.v", 0b100010, 0b00001>;
defm VFCVT_F_XU_V : VALU_FV_VS2<"vfcvt.f.xu.v", 0b100010, 0b00010>;
defm VFCVT_F_X_V : VALU_FV_VS2<"vfcvt.f.x.v", 0b100010, 0b00011>;

// Widening Floating-Point/Integer Type-Convert Instructions
let Constraints = "@earlyclobber $vd", RVVConstraint = WidenCvt in {
defm VFWCVT_XU_F_V : VALU_FV_VS2<"vfwcvt.xu.f.v", 0b100010, 0b01000>;
defm VFWCVT_X_F_V : VALU_FV_VS2<"vfwcvt.x.f.v", 0b100010, 0b01001>;
defm VFWCVT_F_XU_V : VALU_FV_VS2<"vfwcvt.f.xu.v", 0b100010, 0b01010>;
defm VFWCVT_F_X_V : VALU_FV_VS2<"vfwcvt.f.x.v", 0b100010, 0b01011>;
defm VFWCVT_F_F_V : VALU_FV_VS2<"vfwcvt.f.f.v", 0b100010, 0b01100>;
}

// Narrowing Floating-Point/Integer Type-Convert Instructions
let Constraints = "@earlyclobber $vd", RVVConstraint = Narrow in {
defm VFNCVT_XU_F_W : VALU_FV_VS2<"vfncvt.xu.f.w", 0b100010, 0b10000>;
defm VFNCVT_X_F_W : VALU_FV_VS2<"vfncvt.x.f.w", 0b100010, 0b10001>;
defm VFNCVT_F_XU_W : VALU_FV_VS2<"vfncvt.f.xu.w", 0b100010, 0b10010>;
defm VFNCVT_F_X_W : VALU_FV_VS2<"vfncvt.f.x.w", 0b100010, 0b10011>;
defm VFNCVT_F_F_W : VALU_FV_VS2<"vfncvt.f.f.w", 0b100010, 0b10100>;
defm VFNCVT_ROD_F_F_W : VALU_FV_VS2<"vfncvt.rod.f.f.w", 0b100010, 0b10101>;
}

// Vector Single-Width Floating-Point Reduction Instructions
defm VFREDOSUM : VALU_FV_V<"vfredosum", 0b000011>;
defm VFREDSUM : VALU_FV_V<"vfredsum", 0b000001>;
defm VFREDMAX : VALU_FV_V<"vfredmax", 0b000111>;
defm VFREDMIN : VALU_FV_V<"vfredmin", 0b000101>;

nt Reduction Instructions
r $vd" in {
ng instructions for second and mask operands.
// This has the downside that the earlyclobber constraint is too coarse and
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
// will impose unnecessary restrictions by not allowing the destination to
// overlap with the first (wide) operand.
defm VFWREDOSUM : VALU_FV_V<"vfwredosum", 0b110011>;
defm VFWREDSUM : VALU_FV_V<"vfwredsum", 0b110001>;
}

let hasSideEffects = 0, mayLoad = 0, mayStore = 0, vm = 1 in {
// Floating-Point Scalar Move Instructions
def VFMV_F_S : RVInstV<0b010000, 0b00000, OPFVV, (outs FPR32:$vd),
                      (ins VRegOp:$vs2), "vfmv.f.s", "$vd, $vs2">;
def VFMV_S_F : RVInstV2<0b010000, 0b00000, OPFVF, (outs VRegOp:$vd),
                      (ins FPR32:$rs1), "vfmv.s.f", "$vd, $rs1">;

}
} // Predicates = [HasStdExtF, HasStdExtV]
```

llvm/lib/Target/RISCV/RISCVRegisterInfo.td

☰ View Options

Show All 27 Lines

```
class RISCVReg64<RISCVReg32 subreg> : Register<""> {
  let HWEncoding{4-0} = subreg.HWEncoding{4-0};
  let SubRegs = [subreg];
  let SubRegIndices = [sub_32];
  let AsmName = subreg.AsmName;
  let AltNames = subreg.AltNames;
}

class RISCVRegWithSubRegs<bits<5> Enc, string n, list<Register> subregs,
                          list<string> alt = []>
      : RegisterWithSubRegs<n, subregs> {
  let HWEncoding{4-0} = Enc;
  let AltNames = alt;
}

def ABIRegAltName : RegAltNameIndex;

def sub_vrm2    : SubRegIndex<64, -1>;
def sub_vrm2_hi : SubRegIndex<64, -1>;
def sub_vrm4    : SubRegIndex<128, -1>;
def sub_vrm4_hi : SubRegIndex<128, -1>;
def sub_vrm8    : SubRegIndex<256, -1>;
def sub_vrm8_hi : SubRegIndex<256, -1>;
```

fpallares Not Done

My understanding is that here you specify minimum sizes (at least we will have one element of size `ELEN`). As they stand, they are consistent with `ELEN=64`. So far it is unclear to me what sizes we would want to use if we also want to support `ELEN=32`. I imagine we could reuse `HwMode` or something similar.

As far as the offsets are concerned I'd suggest setting them to `-1` because we don't really know the offset where the `hi` part starts.

That said, you already mentioned that types are not that important in this patch.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
// Integer registers
// CostPerUse is set higher for registers that may not be compressible as they
// are not part of GPRC, the most restrictive register class used by the
// compressed instruction set. This will influence the greedy register
// allocator to reduce the use of registers that can't be encoded in 16 bit
// instructions. This affects register allocation even when compressed
```

▲ Show 20 Lines • Show All 183 Lines • ▼ Show 20 Line(s)

```
    (sequence "F%u_D", 8, 9),
    (sequence "F%u_D", 18, 27)
)>;

def FPR64C : RegisterClass<"RISCV", [f64], 64, (add
  (sequence "F%u_D", 10, 15),
  (sequence "F%u_D", 8, 9)
)>;

// Vector registers
let RegAltNameIndices = [ABIRegAltName] in {
```

rogfer01 Not Done

Would it help to simplify this as

```
// Define V0-V31
let RegAltNameIndices = [ABIRegAltName] in {
   foreach Index = 0-31 in {
      def V#Index : RISCVReg<Index, "v"#Index, ["v"#Index]>,
       DwarfRegNum<[!add(#Index, 64)]>;
   }
}
```

```
  foreach Index = 0-31 in {
    def V#Index : RISCVReg<Index, "v"#Index, ["v"#Index]>, DwarfRegNum<[!add(Index, 64)]>;
  }

  foreach Index = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22,
                   24, 26, 28, 30] in {
    def V#Index#M2 : RISCVRegWithSubRegs<Index, "v"#Index,
                       [!cast<Register>("V"#Index),
                        !cast<Register>("V"#!add(Index, 1))],
                       ["v"#Index]>,
                     DwarfRegAlias<!cast<Register>("V"#Index)> {
      let SubRegIndices = [sub_vrm2, sub_vrm2_hi];
    }
  }

  foreach Index = [0, 4, 8, 12, 16, 20, 24, 28] in {
    def V#Index#M4 : RISCVRegWithSubRegs<Index, "v"#Index,
                       [!cast<Register>("V"#Index#"M2"),
                        !cast<Register>("V"#!add(Index, 2)#"M2")],
                       ["v"#Index]>,
                     DwarfRegAlias<!cast<Register>("V"#Index)> {
      let SubRegIndices = [sub_vrm4, sub_vrm4_hi];
    }
  }

                               ] in {
                               thSubRegs<Index, "v"#Index,
                               egister>("V"#Index#"M4"),
                               egister>("V"#!add(Index, 4)#"M4")],
                       ["v"#Index]>,
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
                     DwarfRegAlias<!cast<Register>("V"#Index)> {
      let SubRegIndices = [sub_vrm8, sub_vrm8_hi];
    }
  }

  def VTYPE  : RISCVReg<0, "vtype", ["vtype"]>;
  def VL     : RISCVReg<0, "vl", ["vl"]>;
}

class RegisterTypes<list<ValueType> reg_types> {
  list<ValueType> types = reg_types;
}

```

rogfer01 Not Done

What does the `P` in `VPR` stand for?

```
def VDataVT : RegisterTypes<[nxv8i8, nxv16i8, nxv32i8,
                            nxv4i16, nxv8i16, nxv16i16, nxv32i16,
                            nxv2i32, nxv4i32, nxv8i32, nxv16i32,
                            nxv1i64, nxv2i64, nxv4i64, nxv8i64]>;

// The order of registers represents the preferred allocation sequence,
// meaning caller-save regs are listed before callee-save.
def VR : RegisterClass<"RISCV", [nxv8i8, nxv4i16, nxv2i32, nxv1i64],
                         64, (add
    (sequence "V%u", 25, 31),
    (sequence "V%u", 8, 24),
    (sequence "V%u", 0, 7)
  )> {
  let Size = 64;
}

def VRM2 : RegisterClass<"RISCV", [nxv16i8, nxv8i16, nxv4i32, nxv2i64], 64,
                         (add V26M2, V28M2, V30M2, V8M2, V10M2, V12M2, V14M2, V16M2,
                              V18M2, V20M2, V22M2, V24M2, V0M2, V2M2, V4M2, V6M2)> {
  let Size = 128;
}

def VRM4 : RegisterClass<"RISCV", [nxv32i8, nxv16i16, nxv8i32, nxv4i64], 64,
                         (add V28M4, V8M4, V12M4, V16M4, V20M4, V24M4, V0M4, V4M4)> {
  let Size = 256;
}

def VRM8 : RegisterClass<"RISCV", [nxv32i16, nxv16i32, nxv8i64], 64,
                         (add V8M8, V16M8, V24M8, V0M8)> {
  let Size = 512;
}

def VMaskVT : RegisterTypes<[nxv1i1, nxv2i1, nxv4i1, nxv8i1, nxv16i1, nxv32i1]>;

def VM : RegisterClass<"RISCV", VMaskVT.types, 64, (add
    (sequence "V%u", 25, 31),
    (sequence "V%u", 8, 24),
    (sequence "V%u", 0, 7))> {
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
", VMaskVT.types, 64, (add V0)> {
  let Size = 64;
}
```

llvm/lib/Target/RISCV/RISCVSchedRocket32.td

☰ View Options

Show All 11 Lines

```

// Rocket machine model for scheduling and other instruction cost heuristics.
def Rocket32Model : SchedMachineModel {
  let MicroOpBufferSize = 0; // Explicitly set to zero since Rocket is in-order.
  let IssueWidth = 1;        // 1 micro-ops are dispatched per cycle.
  let LoadLatency = 3;
  let MispredictPenalty = 3;
  let CompleteModel = 1;
  let UnsupportedFeatures = [HasStdExtV];
}

//===----------------------------------------------------------------------===//
// Define each kind of processor resource and number available.

// Modeling each pipeline as a ProcResource using the BufferSize = 0 since
// Rocket is in-order.

```

▲ Show 20 Lines • Show All 199 Lines • Show Last 20 Lines

llvm/lib/Target/RISCV/RISCVSchedRocket64.td

☰ View Options

Show All 10 Lines

```
// This works with MachineScheduler. See MCSchedule.h for details.

// Rocket machine model for scheduling and other instruction cost heuristics.
def Rocket64Model : SchedMachineModel {
  let MicroOpBufferSize = 0; // Explicitly set to zero since Rocket is in-order.
  let IssueWidth = 1;        // 1 micro-ops are dispatched per cycle.
  let LoadLatency = 3;
  let MispredictPenalty = 3;
  let UnsupportedFeatures = [HasStdExtV];
}

//===----------------------------------------------------------------------===//
// Define each kind of processor resource and number available.

// Modeling each pipeline as a ProcResource using the BufferSize = 0 since
// Rocket is in-order.

```

▲ Show 20 Lines • Show All 201 Lines • Show Last 20 Lines

llvm/lib/Target/RISCV/RISCVSubtarget.h

☰ View Options

Show First 20 Lines • Show All 44 Lines • ▼ Show 20 Line(s)

```
  bool HasStdExtZbe = false;
  bool HasStdExtZbf = false;
```

```
  bool HasStdExtZbt = false;
  bool HasStdExtZbproposedc = false;
  bool HasStdExtV = false;
  bool HasRV64 = false;
  bool IsRV32E = false;
  bool EnableLinkerRelax = false;
  bool EnableRVCHintInstrs = false;
  bool EnableSaveRestore = false;
  unsigned XLen = 32;
  MVT XLenVT = MVT::i32;
  RISCVABI::ABI TargetABI = RISCVABI::ABI_Unknown;
```

▲ Show 20 Lines • Show All 44 Lines • ▼ Show 20 Line(s)

```
  bool hasStdExtZbe() const { return HasStdExtZbe; }
  bool hasStdExtZbf() const { return HasStdExtZbf; }
  bool hasStdExtZbm() const { return HasStdExtZbm; }
  bool hasStdExtZbp() const { return HasStdExtZbp; }
  bool hasStdExtZbr() const { return HasStdExtZbr; }
  bool hasStdExtZbs() const { return HasStdExtZbs; }
  bool hasStdExtZbt() const { return HasStdExtZbt; }
  bool hasStdExtZbproposedc() const { return HasStdExtZbproposedc; }
  bool hasStdExtV() const { return HasStdExtV; }
  bool is64Bit() const { return HasRV64; }
  bool isRV32E() const { return IsRV32E; }
  bool enableLinkerRelax() const { return EnableLinkerRelax; }
  bool enableRVCHintInstrs() const { return EnableRVCHintInstrs; }
  bool enableSaveRestore() const { return EnableSaveRestore; }
  MVT getXLenVT() const { return XLenVT; }
  unsigned getXLen() const { return XLen; }
  RISCVABI::ABI getTargetABI() const { return TargetABI; }
```

Show All 21 Lines

🗎 llvm/lib/Target/RISCV/RISCVSystemOperands.td

≡ View Options

Show First 20 Lines • Show All 341 Lines • ▼ Show 20 Line(s)

```
def : SysReg<"tdata3", 0x7A3>;

//===-----------------------------------------------
// Debug Mode Registers
//===-----------------------------------------------
def : SysReg<"dcsr", 0x7B0>;
def : SysReg<"dpc", 0x7B1>;
def : SysReg<"dscratch", 0x7B2>;

//===-----------------------------------------------
// User Vector CSRs
//===-----------------------------------------------
def : SysReg<"vstart", 0x008>;
def : SysReg<"vxsat", 0x009>;
def : SysReg<"vxrm", 0x00A>;
def : SysReg<"vl", 0xC20>;
def : SysReg<"vtype", 0xC21>;
def : SysReg<"vlenb", 0xC22>;
```

🗎 llvm/lib/Target/RISCV/Utils/RISCVBaseInfo.h

≡ View Options

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

Show All 39 Lines

```
  InstFormatCL = 12,
  InstFormatCS = 13,
  InstFormatCA = 14,
  InstFormatCB = 15,
  InstFormatCJ = 16,
  InstFormatOther = 17,

  InstFormatMask = 31
  InstFormatMask = 31,
};

```

⏮ **evandro** Not Done ↩ ✖

Why not just 32 ?

```
// RISC-V Specific Machine Operand Flags
enum {
  MO_None = 0,
  MO_CALL = 1,
  MO_PLT = 2,
  MO_LO = 3,
  MO_HI = 4,
  MO_PCREL_LO = 5,
```

▲ Show 20 Lines • Show All 164 Lines • Show Last 20 Lines

llvm/test/MC/RISCV/rvv/add.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vadd.vv v8, v4, v20, v0.t
# CHECK-INST: vadd.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 00 <unknown>

vadd.vv v8, v4, v20
# CHECK-INST: vadd.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 02 <unknown>

vadd.vx v8, v4, a0, v0.t
# CHECK-INST: vadd.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x00]
                                    ires the following: 'V' (Vector Instructions)
                                    nknown>


# CHECK-INST: vadd.vx v8, v4, a0
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ENCODING: [0x57,0x44,0x45,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 02 <unknown>

vadd.vi v8, v4, 15, v0.t
# CHECK-INST: vadd.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 00 <unknown>

vadd.vi v8, v4, 15
# CHECK-INST: vadd.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 02 <unknown>

vwaddu.vv v8, v4, v20, v0.t
# CHECK-INST: vwaddu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xc0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a c0 <unknown>

vwaddu.vv v8, v4, v20
# CHECK-INST: vwaddu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xc2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a c2 <unknown>

vwaddu.vx v8, v4, a0, v0.t
# CHECK-INST: vwaddu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xc0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 c0 <unknown>

vwaddu.vx v8, v4, a0
# CHECK-INST: vwaddu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xc2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 c2 <unknown>

vwadd.vv v8, v4, v20, v0.t
# CHECK-INST: vwadd.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xc4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a c4 <unknown>

vwadd.vv v8, v4, v20
# CHECK-INST: vwadd.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xc6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a c6 <unknown>

vwadd.vx v8, v4, a0, v0.t
# CHECK-INST: vwadd.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xc4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
                                   nknown>
```

```
                                   a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xc6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 c6 <unknown>

vwaddu.wv v8, v4, v20, v0.t
# CHECK-INST: vwaddu.wv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xd0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a d0 <unknown>

vwaddu.wv v8, v4, v20
# CHECK-INST: vwaddu.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xd2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a d2 <unknown>

vwaddu.wx v8, v4, a0, v0.t
# CHECK-INST: vwaddu.wx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xd0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 d0 <unknown>

vwaddu.wx v8, v4, a0
# CHECK-INST: vwaddu.wx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xd2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 d2 <unknown>

vwadd.wv v8, v4, v20, v0.t
# CHECK-INST: vwadd.wv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xd4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a d4 <unknown>

vwadd.wv v8, v4, v20
# CHECK-INST: vwadd.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xd6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a d6 <unknown>

vwadd.wx v8, v4, a0, v0.t
# CHECK-INST: vwadd.wx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xd4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 d4 <unknown>

vwadd.wx v8, v4, a0
# CHECK-INST: vwadd.wx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xd6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 d6 <unknown>

vadc.vvm v8, v4, v20, v0
# CHECK-INST: vadc.vvm v8, v4, v20, v0
# CHECK-ENCODING: [0x57,0x04,0x4a,0x40]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 40 <unknown>

a0, v0
45,0x40]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 40 <unknown>

vadc.vim v8, v4, 15, v0
# CHECK-INST: vadc.vim v8, v4, 15, v0
# CHECK-ENCODING: [0x57,0xb4,0x47,0x40]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 40 <unknown>

vmadc.vvm v8, v4, v20, v0
# CHECK-INST: vmadc.vvm v8, v4, v20, v0
# CHECK-ENCODING: [0x57,0x04,0x4a,0x44]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 44 <unknown>

vmadc.vxm v8, v4, a0, v0
# CHECK-INST: vmadc.vxm v8, v4, a0, v0
# CHECK-ENCODING: [0x57,0x44,0x45,0x44]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 44 <unknown>

vmadc.vim v8, v4, 15, v0
# CHECK-INST: vmadc.vim v8, v4, 15, v0
# CHECK-ENCODING: [0x57,0xb4,0x47,0x44]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 44 <unknown>

vmadc.vv v8, v4, v20
# CHECK-INST: vmadc.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x46]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 46 <unknown>

vmadc.vx v8, v4, a0
# CHECK-INST: vmadc.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x46]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 46 <unknown>

vmadc.vi v8, v4, 15
# CHECK-INST: vmadc.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x46]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 46 <unknown>

vsaddu.vv v8, v4, v20, v0.t
# CHECK-INST: vsaddu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x80]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 80 <unknown>

vsaddu.vv v8, v4, v20
# CHECK-INST: vsaddu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x82]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 82 <unknown>


                                             a0, v0.t
                                             45,0x80]
                                             ires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 80 <unknown>
```

```

vsaddu.vx v8, v4, a0
# CHECK-INST: vsaddu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x82]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 82 <unknown>

vsaddu.vi v8, v4, 15, v0.t
# CHECK-INST: vsaddu.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x80]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 80 <unknown>

vsaddu.vi v8, v4, 15
# CHECK-INST: vsaddu.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x82]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 82 <unknown>

vsadd.vv v8, v4, v20, v0.t
# CHECK-INST: vsadd.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x84]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 84 <unknown>

vsadd.vv v8, v4, v20
# CHECK-INST: vsadd.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x86]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 86 <unknown>

vsadd.vx v8, v4, a0, v0.t
# CHECK-INST: vsadd.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x84]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 84 <unknown>

vsadd.vx v8, v4, a0
# CHECK-INST: vsadd.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x86]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 86 <unknown>

vsadd.vi v8, v4, 15, v0.t
# CHECK-INST: vsadd.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x84]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 84 <unknown>

vsadd.vi v8, v4, 15
# CHECK-INST: vsadd.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x86]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 86 <unknown>

vaadd.vv v8, v4, v20, v0.t
                               v20, v0.t
                               4a,0x24]
                               ires the following: 'V' (Vector Instructions)
                               nknown>

```

```
vaadd.vv v8, v4, v20
# CHECK-INST: vaadd.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x26]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 26 <unknown>

vaadd.vx v8, v4, a0, v0.t
# CHECK-INST: vaadd.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x24]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 24 <unknown>

vaadd.vx v8, v4, a0
# CHECK-INST: vaadd.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x26]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 26 <unknown>

vaaddu.vv v8, v4, v20, v0.t
# CHECK-INST: vaaddu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x20]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 20 <unknown>

vaaddu.vv v8, v4, v20
# CHECK-INST: vaaddu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x22]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 22 <unknown>

vaaddu.vx v8, v4, a0, v0.t
# CHECK-INST: vaaddu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x20]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 20 <unknown>

vaaddu.vx v8, v4, a0
# CHECK-INST: vaaddu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x22]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 22 <unknown>
```

llvm/test/MC/RISCV/rvv/and.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
20, v0.t
4a,0x24]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 24 <unknown>

vand.vv v8, v4, v20
# CHECK-INST: vand.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x26]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 26 <unknown>

vand.vx v8, v4, a0, v0.t
# CHECK-INST: vand.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x24]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 24 <unknown>

vand.vx v8, v4, a0
# CHECK-INST: vand.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x26]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 26 <unknown>

vand.vi v8, v4, 15, v0.t
# CHECK-INST: vand.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x24]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 24 <unknown>

vand.vi v8, v4, 15
# CHECK-INST: vand.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x26]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 26 <unknown>
```

llvm/test/MC/RISCV/rvv/clip.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vnclipu.wv v8, v4, v20, v0.t
# CHECK-INST: vnclipu.wv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xb8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a b8 <unknown>

vnclipu.wv v8, v4, v20
# CHECK-INST: vnclipu.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0xba]
                              ires the following: 'V' (Vector Instructions)
                              nknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vnclipu.wx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0xb8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 b8 <unknown>

vnclipu.wx v8, v4, a0
# CHECK-INST: vnclipu.wx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0xba]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 ba <unknown>

vnclipu.wi v8, v4, 31, v0.t
# CHECK-INST: vnclipu.wi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xb8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f b8 <unknown>

vnclipu.wi v8, v4, 31
# CHECK-INST: vnclipu.wi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xba]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f ba <unknown>

vnclip.wv v8, v4, v20, v0.t
# CHECK-INST: vnclip.wv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xbc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a bc <unknown>

vnclip.wv v8, v4, v20
# CHECK-INST: vnclip.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0xbe]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a be <unknown>

vnclip.wx v8, v4, a0, v0.t
# CHECK-INST: vnclip.wx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0xbc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 bc <unknown>

vnclip.wx v8, v4, a0
# CHECK-INST: vnclip.wx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0xbe]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 be <unknown>

vnclip.wi v8, v4, 31, v0.t
# CHECK-INST: vnclip.wi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xbc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f bc <unknown>

vnclip.wi v8, v4, 31
# CHECK-INST: vnclip.wi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xbe]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f be <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vmseq.vv v8, v4, v20, v0.t
# CHECK-INST: vmseq.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x60]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 60 <unknown>

vmseq.vv v8, v4, v20
# CHECK-INST: vmseq.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x62]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 62 <unknown>

vmseq.vx v8, v4, a0, v0.t
# CHECK-INST: vmseq.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x60]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 60 <unknown>

vmseq.vx v8, v4, a0
# CHECK-INST: vmseq.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x62]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 62 <unknown>

vmseq.vi v8, v4, 15, v0.t
# CHECK-INST: vmseq.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x60]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 60 <unknown>

vmseq.vi v8, v4, 15
# CHECK-INST: vmseq.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x62]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 62 <unknown>

vmsne.vv v8, v4, v20, v0.t
# CHECK-INST: vmsne.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x64]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 64 <unknown>

vmsne.vv v8, v4, v20
# CHECK-INST: vmsne.vv v8, v4, v20
                                     4a,0x66]
                                     ires the following: 'V' (Vector Instructions)
                                     nknown>

vmsne.vx v8, v4, a0, v0.t
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vmsne.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x64]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 64 <unknown>

vmsne.vx v8, v4, a0
# CHECK-INST: vmsne.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x66]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 66 <unknown>

vmsne.vi v8, v4, 15, v0.t
# CHECK-INST: vmsne.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x64]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 64 <unknown>

vmsne.vi v8, v4, 15
# CHECK-INST: vmsne.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x66]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 66 <unknown>

vmsltu.vv v8, v4, v20, v0.t
# CHECK-INST: vmsltu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x68]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 68 <unknown>

vmsltu.vv v8, v4, v20
# CHECK-INST: vmsltu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x6a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 6a <unknown>

vmsltu.vx v8, v4, a0, v0.t
# CHECK-INST: vmsltu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x68]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 68 <unknown>

vmsltu.vx v8, v4, a0
# CHECK-INST: vmsltu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x6a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 6a <unknown>

vmslt.vv v8, v4, v20, v0.t
# CHECK-INST: vmslt.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x6c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 6c <unknown>

vmslt.vv v8, v4, v20
# CHECK-INST: vmslt.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x6e]
                            ires the following: 'V' (Vector Instructions)
                            nknown>


# CHECK-INST: vmslt.vx v8, v4, a0, v0.t
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ENCODING: [0x57,0x44,0x45,0x6c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 6c <unknown>

vmslt.vx v8, v4, a0
# CHECK-INST: vmslt.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x6e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 6e <unknown>

vmsleu.vv v8, v4, v20, v0.t
# CHECK-INST: vmsleu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x70]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 70 <unknown>

vmsleu.vv v8, v4, v20
# CHECK-INST: vmsleu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x72]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 72 <unknown>

vmsleu.vx v8, v4, a0, v0.t
# CHECK-INST: vmsleu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x70]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 70 <unknown>

vmsleu.vx v8, v4, a0
# CHECK-INST: vmsleu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x72]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 72 <unknown>

vmsleu.vi v8, v4, 15, v0.t
# CHECK-INST: vmsleu.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x70]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 70 <unknown>

vmsleu.vi v8, v4, 15
# CHECK-INST: vmsleu.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x72]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 72 <unknown>

vmsle.vv v8, v4, v20, v0.t
# CHECK-INST: vmsle.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x74]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 74 <unknown>

vmsle.vv v8, v4, v20
# CHECK-INST: vmsle.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x76]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 76 <unknown>


a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x74]
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 74 <unknown>

vmsle.vx v8, v4, a0
# CHECK-INST: vmsle.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x76]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 76 <unknown>

vmsle.vi v8, v4, 15, v0.t
# CHECK-INST: vmsle.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x74]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 74 <unknown>

vmsle.vi v8, v4, 15
# CHECK-INST: vmsle.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x76]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 76 <unknown>

vmsgtu.vx v8, v4, a0, v0.t
# CHECK-INST: vmsgtu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x78]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 78 <unknown>

vmsgtu.vx v8, v4, a0
# CHECK-INST: vmsgtu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x7a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 7a <unknown>

vmsgtu.vi v8, v4, 15, v0.t
# CHECK-INST: vmsgtu.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x78]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 78 <unknown>

vmsgtu.vi v8, v4, 15
# CHECK-INST: vmsgtu.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x7a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 7a <unknown>

vmsgt.vx v8, v4, a0, v0.t
# CHECK-INST: vmsgt.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x7c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 7c <unknown>

vmsgt.vx v8, v4, a0
# CHECK-INST: vmsgt.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x7e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 7e <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
15, v0.t
47,0x7c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 7c <unknown>

vmsgt.vi v8, v4, 15
# CHECK-INST: vmsgt.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x7e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 7e <unknown>
```

llvm/test/MC/RISCV/rvv/convert.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfcvt.xu.f.v v8, v4, v0.t
# CHECK-INST: vfcvt.xu.f.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x40,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 40 88 <unknown>

vfcvt.xu.f.v v8, v4
# CHECK-INST: vfcvt.xu.f.v v8, v4
# CHECK-ENCODING: [0x57,0x14,0x40,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 40 8a <unknown>

vfcvt.x.f.v v8, v4, v0.t
# CHECK-INST: vfcvt.x.f.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x94,0x40,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 40 88 <unknown>

vfcvt.x.f.v v8, v4
# CHECK-INST: vfcvt.x.f.v v8, v4
# CHECK-ENCODING: [0x57,0x94,0x40,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 40 8a <unknown>

vfcvt.f.xu.v v8, v4, v0.t
# CHECK-INST: vfcvt.f.xu.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x41,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 41 88 <unknown>

                                  v4
                                  41,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
(Vector Instructions)
# CHECK-UNKNOWN: 57 14 41 8a <unknown>

vfcvt.f.x.v v8, v4, v0.t
# CHECK-INST: vfcvt.f.x.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x94,0x41,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 41 88 <unknown>

vfcvt.f.x.v v8, v4
# CHECK-INST: vfcvt.f.x.v v8, v4
# CHECK-ENCODING: [0x57,0x94,0x41,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 41 8a <unknown>

vfwcvt.xu.f.v v8, v4, v0.t
# CHECK-INST: vfwcvt.xu.f.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x44,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 44 88 <unknown>

vfwcvt.xu.f.v v8, v4
# CHECK-INST: vfwcvt.xu.f.v v8, v4
# CHECK-ENCODING: [0x57,0x14,0x44,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 44 8a <unknown>

vfwcvt.x.f.v v8, v4, v0.t
# CHECK-INST: vfwcvt.x.f.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x94,0x44,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 44 88 <unknown>

vfwcvt.x.f.v v8, v4
# CHECK-INST: vfwcvt.x.f.v v8, v4
# CHECK-ENCODING: [0x57,0x94,0x44,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 44 8a <unknown>

vfwcvt.f.xu.v v8, v4, v0.t
# CHECK-INST: vfwcvt.f.xu.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x45,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 45 88 <unknown>

vfwcvt.f.xu.v v8, v4
# CHECK-INST: vfwcvt.f.xu.v v8, v4
# CHECK-ENCODING: [0x57,0x14,0x45,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
                              nknown>



                              v4, v0.t
# CHECK-ENCODING: [0x57,0x94,0x45,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 45 88 <unknown>

vfwcvt.f.x.v v8, v4
# CHECK-INST: vfwcvt.f.x.v v8, v4
# CHECK-ENCODING: [0x57,0x94,0x45,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 45 8a <unknown>

vfwcvt.f.f.v v8, v4, v0.t
# CHECK-INST: vfwcvt.f.f.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x46,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 46 88 <unknown>

vfwcvt.f.f.v v8, v4
# CHECK-INST: vfwcvt.f.f.v v8, v4
# CHECK-ENCODING: [0x57,0x14,0x46,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 46 8a <unknown>

vfncvt.xu.f.w v8, v4, v0.t
# CHECK-INST: vfncvt.xu.f.w v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x48,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 48 88 <unknown>

vfncvt.xu.f.w v8, v4
# CHECK-INST: vfncvt.xu.f.w v8, v4
# CHECK-ENCODING: [0x57,0x14,0x48,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 48 8a <unknown>

vfncvt.x.f.w v8, v4, v0.t
# CHECK-INST: vfncvt.x.f.w v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x94,0x48,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 48 88 <unknown>

vfncvt.x.f.w v8, v4
# CHECK-INST: vfncvt.x.f.w v8, v4
# CHECK-ENCODING: [0x57,0x94,0x48,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 48 8a <unknown>

vfncvt.f.xu.w v8, v4, v0.t
# CHECK-INST: vfncvt.f.xu.w v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x49,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
nknown>
```

```
# CHECK-INST: vfncvt.f.xu.w v8, v4
# CHECK-ENCODING: [0x57,0x14,0x49,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 49 8a <unknown>

vfncvt.f.x.w v8, v4, v0.t
# CHECK-INST: vfncvt.f.x.w v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x94,0x49,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 49 88 <unknown>

vfncvt.f.x.w v8, v4
# CHECK-INST: vfncvt.f.x.w v8, v4
# CHECK-ENCODING: [0x57,0x94,0x49,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 49 8a <unknown>

vfncvt.f.f.w v8, v4, v0.t
# CHECK-INST: vfncvt.f.f.w v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 88 <unknown>

vfncvt.f.f.w v8, v4
# CHECK-INST: vfncvt.f.f.w v8, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 8a <unknown>

vfncvt.rod.f.f.w v8, v4, v0.t
# CHECK-INST: vfncvt.rod.f.f.w v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x94,0x4a,0x88]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 4a 88 <unknown>

vfncvt.rod.f.f.w v8, v4
# CHECK-INST: vfncvt.rod.f.f.w v8, v4
# CHECK-ENCODING: [0x57,0x94,0x4a,0x8a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 94 4a 8a <unknown>
```

llvm/test/MC/RISCV/rvv/div.s

View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
                                  --mattr=+experimental-v - \
                                 check-prefix=CHECK-INST
                                 -filetype=obj --mattr=+experimental-v < %s \
                                  - | FileCheck %s --check-prefix=CHECK-UNKNOWN

```

```
vdivu.vv v8, v4, v20, v0.t
# CHECK-INST: vdivu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x80]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 80 <unknown>

vdivu.vv v8, v4, v20
# CHECK-INST: vdivu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x82]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 82 <unknown>

vdivu.vx v8, v4, a0, v0.t
# CHECK-INST: vdivu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x80]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 80 <unknown>

vdivu.vx v8, v4, a0
# CHECK-INST: vdivu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x82]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 82 <unknown>

vdiv.vv v8, v4, v20, v0.t
# CHECK-INST: vdiv.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x84]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 84 <unknown>

vdiv.vv v8, v4, v20
# CHECK-INST: vdiv.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x86]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 86 <unknown>

vdiv.vx v8, v4, a0, v0.t
# CHECK-INST: vdiv.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x84]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 84 <unknown>

vdiv.vx v8, v4, a0
# CHECK-INST: vdiv.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x86]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 86 <unknown>

vremu.vv v8, v4, v20, v0.t
# CHECK-INST: vremu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x88]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 88 <unknown>

vremu.vv v8, v4, v20
# CHECK-INST: vremu.vv v8, v4, v20
                                   4a,0x8a]
                                   ires the following: 'V' (Vector Instructions)
                                   nknown>

vremu.vx v8, v4, a0, v0.t
```

```
# CHECK-INST: vremu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x88]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 88 <unknown>

vremu.vx v8, v4, a0
# CHECK-INST: vremu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x8a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 8a <unknown>

vrem.vv v8, v4, v20, v0.t
# CHECK-INST: vrem.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x8c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 8c <unknown>

vrem.vv v8, v4, v20
# CHECK-INST: vrem.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x8e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 8e <unknown>

vrem.vx v8, v4, a0, v0.t
# CHECK-INST: vrem.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x8c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 8c <unknown>

vrem.vx v8, v4, a0
# CHECK-INST: vrem.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x8e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 8e <unknown>
```

llvm/test/MC/RISCV/rvv/fadd.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfadd.vv v8, v4, v20, v0.t
# CHECK-INST: vfadd.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x00]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 00 <unknown>


                                                v20
                                                4a,0x02]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
(Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 02 <unknown>

vfadd.vf v8, v4, fa0, v0.t
# CHECK-INST: vfadd.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x00]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 00 <unknown>

vfadd.vf v8, v4, fa0
# CHECK-INST: vfadd.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x02]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 02 <unknown>

vfwadd.vv v8, v4, v20, v0.t
# CHECK-INST: vfwadd.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xc0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a c0 <unknown>

vfwadd.vv v8, v4, v20
# CHECK-INST: vfwadd.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0xc2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a c2 <unknown>

vfwadd.vf v8, v4, fa0, v0.t
# CHECK-INST: vfwadd.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xc0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 c0 <unknown>

vfwadd.vf v8, v4, fa0
# CHECK-INST: vfwadd.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0xc2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 c2 <unknown>

vfwadd.wv v8, v4, v20, v0.t
# CHECK-INST: vfwadd.wv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xd0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a d0 <unknown>

vfwadd.wv v8, v4, v20
# CHECK-INST: vfwadd.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0xd2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
nknown>


fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xd0]
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 d0 <unknown>

vfwadd.wf v8, v4, fa0
# CHECK-INST: vfwadd.wf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0xd2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 d2 <unknown>
```

llvm/test/MC/RISCV/rvv/fcompare.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vmfeq.vv v8, v4, v20, v0.t
# CHECK-INST: vmfeq.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x60]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 60 <unknown>

vmfeq.vv v8, v4, v20
# CHECK-INST: vmfeq.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x62]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 62 <unknown>

vmfeq.vf v8, v4, fa0, v0.t
# CHECK-INST: vmfeq.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x60]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 60 <unknown>

vmfeq.vf v8, v4, fa0
# CHECK-INST: vmfeq.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x62]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 62 <unknown>

vmfne.vv v8, v4, v20, v0.t
# CHECK-INST: vmfne.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x70]
ires the following: 'F' (Single-Precision Floating-Point), 'V'
nknown>

vmfne.vv v8, v4, v20
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vmfne.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x72]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 72 <unknown>

vmfne.vf v8, v4, fa0, v0.t
# CHECK-INST: vmfne.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x70]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 70 <unknown>

vmfne.vf v8, v4, fa0
# CHECK-INST: vmfne.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x72]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 72 <unknown>

vmflt.vv v8, v4, v20, v0.t
# CHECK-INST: vmflt.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x6c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 6c <unknown>

vmflt.vv v8, v4, v20
# CHECK-INST: vmflt.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x6e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 6e <unknown>

vmflt.vf v8, v4, fa0, v0.t
# CHECK-INST: vmflt.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x6c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 6c <unknown>

vmflt.vf v8, v4, fa0
# CHECK-INST: vmflt.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x6e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 6e <unknown>

vmfle.vv v8, v4, v20, v0.t
# CHECK-INST: vmfle.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x64]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 64 <unknown>

vmfle.vv v8, v4, v20
# CHECK-INST: vmfle.vv v8, v4, v20
4a,0x66]
ires the following: 'F' (Single-Precision Floating-Point), 'V'

nknown>

```

```
vmfle.vf v8, v4, fa0, v0.t
# CHECK-INST: vmfle.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x64]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 64 <unknown>

vmfle.vf v8, v4, fa0
# CHECK-INST: vmfle.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x66]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 66 <unknown>

vmfgt.vf v8, v4, fa0, v0.t
# CHECK-INST: vmfgt.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x74]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 74 <unknown>

vmfgt.vf v8, v4, fa0
# CHECK-INST: vmfgt.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x76]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 76 <unknown>

vmfge.vf v8, v4, fa0, v0.t
# CHECK-INST: vmfge.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x7c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 7c <unknown>

vmfge.vf v8, v4, fa0
# CHECK-INST: vmfge.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x7e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 7e <unknown>
```

llvm/test/MC/RISCV/rvv/fdiv.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

v20, v0.t
4a,0x80]
ires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-UNKNOWN: 57 14 4a 80 <unknown>

vfdiv.vv v8, v4, v20
# CHECK-INST: vfdiv.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x82]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 82 <unknown>

vfdiv.vf v8, v4, fa0, v0.t
# CHECK-INST: vfdiv.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x80]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 80 <unknown>

vfdiv.vf v8, v4, fa0
# CHECK-INST: vfdiv.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x82]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 82 <unknown>

vfrdiv.vf v8, v4, fa0, v0.t
# CHECK-INST: vfrdiv.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x84]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 84 <unknown>

vfrdiv.vf v8, v4, fa0
# CHECK-INST: vfrdiv.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x86]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 86 <unknown>
```

llvm/test/MC/RISCV/rvv/fmacc.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfmacc.vv v8, v20, v4, v0.t
# CHECK-INST: vfmacc.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xb0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
nknown>


, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xb2]
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a b2 <unknown>

vfmacc.vf v8, fa0, v4, v0.t
# CHECK-INST: vfmacc.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xb0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 b0 <unknown>

vfmacc.vf v8, fa0, v4
# CHECK-INST: vfmacc.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xb2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 b2 <unknown>

vfnmacc.vv v8, v20, v4, v0.t
# CHECK-INST: vfnmacc.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xb4]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a b4 <unknown>

vfnmacc.vv v8, v20, v4
# CHECK-INST: vfnmacc.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xb6]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a b6 <unknown>

vfnmacc.vf v8, fa0, v4, v0.t
# CHECK-INST: vfnmacc.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xb4]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 b4 <unknown>

vfnmacc.vf v8, fa0, v4
# CHECK-INST: vfnmacc.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xb6]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 b6 <unknown>

vfmsac.vv v8, v20, v4, v0.t
# CHECK-INST: vfmsac.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xb8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a b8 <unknown>

vfmsac.vv v8, v20, v4
# CHECK-INST: vfmsac.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xba]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
nknown>

# CHECK-INST: vfmsac.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xb8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 b8 <unknown>

vfmsac.vf v8, fa0, v4
# CHECK-INST: vfmsac.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xba]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 ba <unknown>

vfnmsac.vv v8, v20, v4, v0.t
# CHECK-INST: vfnmsac.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xbc]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a bc <unknown>

vfnmsac.vv v8, v20, v4
# CHECK-INST: vfnmsac.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xbe]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a be <unknown>

vfnmsac.vf v8, fa0, v4, v0.t
# CHECK-INST: vfnmsac.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xbc]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 bc <unknown>

vfnmsac.vf v8, fa0, v4
# CHECK-INST: vfnmsac.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xbe]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 be <unknown>

vfmadd.vv v8, v20, v4, v0.t
# CHECK-INST: vfmadd.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xa0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a a0 <unknown>

vfmadd.vv v8, v20, v4
# CHECK-INST: vfmadd.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xa2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a a2 <unknown>

vfmadd.vf v8, fa0, v4, v0.t
# CHECK-INST: vfmadd.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xa0]
                    ires the following: 'F' (Single-Precision Floating-Point), 'V'

                    nknown>

vfmadd.vf v8, fa0, v4
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vfmadd.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xa2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 a2 <unknown>

vfnmadd.vv v8, v20, v4, v0.t
# CHECK-INST: vfnmadd.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xa4]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a a4 <unknown>

vfnmadd.vv v8, v20, v4
# CHECK-INST: vfnmadd.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xa6]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a a6 <unknown>

vfnmadd.vf v8, fa0, v4, v0.t
# CHECK-INST: vfnmadd.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xa4]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 a4 <unknown>

vfnmadd.vf v8, fa0, v4
# CHECK-INST: vfnmadd.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xa6]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 a6 <unknown>

vfmsub.vv v8, v20, v4, v0.t
# CHECK-INST: vfmsub.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xa8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a a8 <unknown>

vfmsub.vv v8, v20, v4
# CHECK-INST: vfmsub.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xaa]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a aa <unknown>

vfmsub.vf v8, fa0, v4, v0.t
# CHECK-INST: vfmsub.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xa8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 a8 <unknown>

vfmsub.vf v8, fa0, v4
# CHECK-INST: vfmsub.vf v8, fa0, v4
45,0xaa]
ires the following: 'F' (Single-Precision Floating-Point), 'V'
nknown>

```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
vfnmsub.vv v8, v20, v4, v0.t
# CHECK-INST: vfnmsub.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xac]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a ac <unknown>

vfnmsub.vv v8, v20, v4
# CHECK-INST: vfnmsub.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xae]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a ae <unknown>

vfnmsub.vf v8, fa0, v4, v0.t
# CHECK-INST: vfnmsub.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xac]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 ac <unknown>

vfnmsub.vf v8, fa0, v4
# CHECK-INST: vfnmsub.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xae]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 ae <unknown>

vfwmacc.vv v8, v20, v4, v0.t
# CHECK-INST: vfwmacc.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xf0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a f0 <unknown>

vfwmacc.vv v8, v20, v4
# CHECK-INST: vfwmacc.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xf2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a f2 <unknown>

vfwmacc.vf v8, fa0, v4, v0.t
# CHECK-INST: vfwmacc.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xf0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 f0 <unknown>

vfwmacc.vf v8, fa0, v4
# CHECK-INST: vfwmacc.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xf2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 f2 <unknown>

vfwnmacc.vv v8, v20, v4, v0.t
20, v4, v0.t
4a,0xf4]
ires the following: 'F' (Single-Precision Floating-Point), 'V'
# CHECK-UNKNOWN: 57 14 4a f4 <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```

vfwnmacc.vv v8, v20, v4
# CHECK-INST: vfwnmacc.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xf6]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a f6 <unknown>

vfwnmacc.vf v8, fa0, v4, v0.t
# CHECK-INST: vfwnmacc.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xf4]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 f4 <unknown>

vfwnmacc.vf v8, fa0, v4
# CHECK-INST: vfwnmacc.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xf6]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 f6 <unknown>

vfwmsac.vv v8, v20, v4, v0.t
# CHECK-INST: vfwmsac.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xf8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a f8 <unknown>

vfwmsac.vv v8, v20, v4
# CHECK-INST: vfwmsac.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x14,0x4a,0xfa]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a fa <unknown>

vfwmsac.vf v8, fa0, v4, v0.t
# CHECK-INST: vfwmsac.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xf8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 f8 <unknown>

vfwmsac.vf v8, fa0, v4
# CHECK-INST: vfwmsac.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xfa]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 fa <unknown>

vfwnmsac.vv v8, v20, v4, v0.t
# CHECK-INST: vfwnmsac.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xfc]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a fc <unknown>

                                   20, v4
                                   4a,0xfe]
                                   ires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-UNKNOWN: 57 14 4a fe <unknown>

vfwnmsac.vf v8, fa0, v4, v0.t
# CHECK-INST: vfwnmsac.vf v8, fa0, v4, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xfc]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 fc <unknown>

vfwnmsac.vf v8, fa0, v4
# CHECK-INST: vfwnmsac.vf v8, fa0, v4
# CHECK-ENCODING: [0x57,0x54,0x45,0xfe]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 fe <unknown>
```

llvm/test/MC/RISCV/rvv/fminmax.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfmin.vv v8, v4, v20, v0.t
# CHECK-INST: vfmin.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x10]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 10 <unknown>

vfmin.vv v8, v4, v20
# CHECK-INST: vfmin.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x12]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 12 <unknown>

vfmin.vf v8, v4, fa0, v0.t
# CHECK-INST: vfmin.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x10]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 10 <unknown>

vfmin.vf v8, v4, fa0
# CHECK-INST: vfmin.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x12]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
nknown>



v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x18]
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 18 <unknown>

vfmax.vv v8, v4, v20
# CHECK-INST: vfmax.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x1a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 1a <unknown>

vfmax.vf v8, v4, fa0, v0.t
# CHECK-INST: vfmax.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x18]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 18 <unknown>

vfmax.vf v8, v4, fa0
# CHECK-INST: vfmax.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x1a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 1a <unknown>
```

llvm/test/MC/RISCV/rvv/fmul.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfmul.vv v8, v4, v20, v0.t
# CHECK-INST: vfmul.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x90]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 90 <unknown>

vfmul.vv v8, v4, v20
# CHECK-INST: vfmul.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x92]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 92 <unknown>

vfmul.vf v8, v4, fa0, v0.t
# CHECK-INST: vfmul.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x90]
                                    ires the following: 'F' (Single-Precision Floating-Point), 'V'

                                    nknown>

vfmul.vf v8, v4, fa0
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vfmul.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x92]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 92 <unknown>

vfwmul.vv v8, v4, v20, v0.t
# CHECK-INST: vfwmul.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xe0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a e0 <unknown>

vfwmul.vv v8, v4, v20
# CHECK-INST: vfwmul.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0xe2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a e2 <unknown>

vfwmul.vf v8, v4, fa0, v0.t
# CHECK-INST: vfwmul.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xe0]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 e0 <unknown>

vfwmul.vf v8, v4, fa0
# CHECK-INST: vfwmul.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0xe2]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 e2 <unknown>
```

llvm/test/MC/RISCV/rvv/fmv.s

View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfmv.v.f v8, fa0
# CHECK-INST: vfmv.v.f v8, fa0
# CHECK-ENCODING: [0x57,0x54,0x05,0x5e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 05 5e <unknown>

vfmv.f.s fa0, v4
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
40,0x42]
ires the following: 'F' (Single-Precision Floating-Point), 'V'
# CHECK-UNKNOWN: 57 15 40 42 <unknown>
```

```

vfmv.s.f v8, fa0
# CHECK-INST: vfmv.s.f v8, fa0
# CHECK-ENCODING: [0x57,0x54,0x05,0x42]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 05 42 <unknown>
```

llvm/test/MC/RISCV/rvv/fothers.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfsqrt.v v8, v4, v0.t
# CHECK-INST: vfsqrt.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x40,0x8c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 40 8c <unknown>

vfsqrt.v v8, v4
# CHECK-INST: vfsqrt.v v8, v4
# CHECK-ENCODING: [0x57,0x14,0x40,0x8e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 40 8e <unknown>

vfclass.v v8, v4, v0.t
# CHECK-INST: vfclass.v v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x14,0x48,0x8c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 48 8c <unknown>

vfclass.v v8, v4
# CHECK-INST: vfclass.v v8, v4
# CHECK-ENCODING: [0x57,0x14,0x48,0x8e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 48 8e <unknown>

vfmerge.vfm v8, v4, fa0, v0
# CHECK-INST: vfmerge.vfm v8, v4, fa0, v0
# CHECK-ENCODING: [0x57,0x54,0x45,0x5c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 5c <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

n.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfredosum.vs v8, v4, v20, v0.t
# CHECK-INST: vfredosum.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x0c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 0c <unknown>

vfredosum.vs v8, v4, v20
# CHECK-INST: vfredosum.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x0e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 0e <unknown>

vfredsum.vs v8, v4, v20, v0.t
# CHECK-INST: vfredsum.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x04]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 04 <unknown>

vfredsum.vs v8, v4, v20
# CHECK-INST: vfredsum.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x06]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 06 <unknown>

vfredmax.vs v8, v4, v20, v0.t
# CHECK-INST: vfredmax.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x1c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 1c <unknown>

vfredmax.vs v8, v4, v20
# CHECK-INST: vfredmax.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x1e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 1e <unknown>

vfredmin.vs v8, v4, v20, v0.t
# CHECK-INST: vfredmin.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x14]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
nknown>

# CHECK-INST: vfredmin.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x16]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 16 <unknown>

vfwredosum.vs v8, v4, v20, v0.t
# CHECK-INST: vfwredosum.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xcc]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a cc <unknown>

vfwredosum.vs v8, v4, v20
# CHECK-INST: vfwredosum.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0xce]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a ce <unknown>

vfwredsum.vs v8, v4, v20, v0.t
# CHECK-INST: vfwredsum.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xc4]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a c4 <unknown>

vfwredsum.vs v8, v4, v20
# CHECK-INST: vfwredsum.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0xc6]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a c6 <unknown>
```

llvm/test/MC/RISCV/rvv/fsub.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfsub.vv v8, v4, v20, v0.t
# CHECK-INST: vfsub.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x08]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 08 <unknown>

vfsub.vv v8, v4, v20
# CHECK-INST: vfsub.vv v8, v4, v20
4a,0x0a]
ires the following: 'F' (Single-Precision Floating-Point), 'V'

nknown>

```

```
vfsub.vf v8, v4, fa0, v0.t
# CHECK-INST: vfsub.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x08]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 08 <unknown>

vfsub.vf v8, v4, fa0
# CHECK-INST: vfsub.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x0a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 0a <unknown>

vfrsub.vf v8, v4, fa0, v0.t
# CHECK-INST: vfrsub.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x9c]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 9c <unknown>

vfrsub.vf v8, v4, fa0
# CHECK-INST: vfrsub.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x9e]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 9e <unknown>

vfwsub.vv v8, v4, v20, v0.t
# CHECK-INST: vfwsub.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0xc8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a c8 <unknown>

vfwsub.vv v8, v4, v20
# CHECK-INST: vfwsub.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0xca]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a ca <unknown>

vfwsub.vf v8, v4, fa0, v0.t
# CHECK-INST: vfwsub.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xc8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 c8 <unknown>

vfwsub.vf v8, v4, fa0
# CHECK-INST: vfwsub.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0xca]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 ca <unknown>

vfwsub.wv v8, v4, v20, v0.t
                               v20, v0.t
                               4a,0xd8]
                               ires the following: 'F' (Single-Precision Floating-Point), 'V'

# CHECK-UNKNOWN: 57 14 4a d8 <unknown>
```

```

vfwsub.wv v8, v4, v20
# CHECK-INST: vfwsub.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0xda]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a da <unknown>

vfwsub.wf v8, v4, fa0, v0.t
# CHECK-INST: vfwsub.wf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0xd8]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 d8 <unknown>

vfwsub.wf v8, v4, fa0
# CHECK-INST: vfwsub.wf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0xda]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 da <unknown>
```

🗎 llvm/test/MC/RISCV/rvv/invalid.s

≡ View Options

This file was added.

```
# RUN: not llvm-mc -triple=riscv64 --mattr=+experimental-v --mattr=+f < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR

vsetvli a2, a0, e31
# CHECK-ERROR: operand must be e[8|16|32|64|128|256|512|1024],m[1|2|4|8]

```

⏮ **fpallares** Not Done ↩ ✖

Suggestion: If you use `CHECK-LABEL` between the regular `CHECK` you can avoid these directives to succeed by matching lines corresponding to a later instruction. Like this:

```
vsetvli a2, a0, e31
// CHECK-ERROR: operand must be e[8|16|32|64|128|256|512|1024],m[1|2|4|8]
// CHECK-ERROR-LABEL: vsetvli a2, a0, e31
```

(Note the `CHECK-LABEL` is placed *after* the `CHECK` since it appears in this order in the output)

Without the `CHECK-LABELS` the test will still fail when something is wrong, but with them FileCheck will be able to point exactly where the problem occurred.

More info on this in the corresponding section of the FileCheck manual.

⏮ **HsiangKai** Author ✓ Done ↩ ✖

Got it. Thanks.

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
e[8|16|32|64|128|256|512|1024],m[1|2|4|8]
vsetvli a2, a0, m1,e32
```

```
# CHECK-ERROR: operand must be e[8|16|32|64|128|256|512|1024],m[1|2|4|8]

vsetvli a2, a0, e32,m16
# CHECK-ERROR: operand must be e[8|16|32|64|128|256|512|1024],m[1|2|4|8]

vsetvli a2, a0, e2048,m8
# CHECK-ERROR: operand must be e[8|16|32|64|128|256|512|1024],m[1|2|4|8]

vsetvli a2, a0, e1,m8
# CHECK-ERROR: operand must be e[8|16|32|64|128|256|512|1024],m[1|2|4|8]

vadd.vv v1, v3, v2, v4.t
# CHECK-ERROR: operand must be v0.t

vadd.vv v1, v3, v2, v0
# CHECK-ERROR: expected '.t' suffix

viota.m v0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: viota.m v0, v2, v0.t

viota.m v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: viota.m v2, v2

vfwcvt.xu.f.v v0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwcvt.xu.f.v v0, v2, v0.t

vfwcvt.xu.f.v v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwcvt.xu.f.v v2, v2

vfwcvt.x.f.v v0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwcvt.x.f.v v0, v2, v0.t

vfwcvt.x.f.v v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwcvt.x.f.v v2, v2

vfwcvt.f.xu.v v0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwcvt.f.xu.v v0, v2, v0.t

vfwcvt.f.xu.v v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwcvt.f.xu.v v2, v2

vfwcvt.f.x.v v0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwcvt.f.x.v v0, v2, v0.t


vector register group cannot overlap the source vector registe

.v v2, v2

```

```
vfwcvt.f.f.v v0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwcvt.f.f.v v0, v2, v0.t

vfwcvt.f.f.v v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwcvt.f.f.v v2, v2

vslideup.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vslideup.vx v0, v2, a0, v0.t

vslideup.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vslideup.vx v2, v2, a0

vslideup.vi v0, v2, 31, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vslideup.vi v0, v2, 31, v0.t

vslideup.vi v2, v2, 31
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vslideup.vi v2, v2, 31

vslide1up.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vslide1up.vx v0, v2, a0, v0.t

vslide1up.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vslide1up.vx v2, v2, a0

vnsrl.wv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnsrl.wv v2, v2, v4

vnsrl.wx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnsrl.wx v2, v2, a0

vnsrl.wi v2, v2, 31
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnsrl.wi v2, v2, 31

vnsra.wv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnsra.wv v2, v2, v4

vnsra.wx v2, v2, a0
vector register group cannot overlap the source vector registe
2, v2, a0

vnsra.wi v2, v2, 31
```

```
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnsra.wi v2, v2, 31

vnclipu.wv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnclipu.wv v2, v2, v4

vnclipu.wx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnclipu.wx v2, v2, a0

vnclipu.wi v2, v2, 31
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnclipu.wi v2, v2, 31

vnclip.wv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnclip.wv v2, v2, v4

vnclip.wx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnclip.wx v2, v2, a0

vnclip.wi v2, v2, 31
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vnclip.wi v2, v2, 31

vfncvt.xu.f.w v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfncvt.xu.f.w v2, v2

vfncvt.x.f.w v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfncvt.x.f.w v2, v2

vfncvt.f.xu.w v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfncvt.f.xu.w v2, v2

vfncvt.f.x.w v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfncvt.f.x.w v2, v2

vfncvt.f.f.w v2, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfncvt.f.f.w v2, v2

vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfncvt.rod.f.f.w v2, v2
```

```

vrgather.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vrgather.vv v0, v2, v4, v0.t

vrgather.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vrgather.vv v2, v2, v4

vrgather.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vrgather.vx v0, v2, a0, v0.t

vrgather.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vrgather.vx v2, v2, a0

vrgather.vi v0, v2, 31, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vrgather.vi v0, v2, 31, v0.t

vrgather.vi v2, v2, 31
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vrgather.vi v2, v2, 31

vwaddu.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwaddu.vv v0, v2, v4, v0.t

vwaddu.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwaddu.vv v2, v2, v4

vwsubu.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwsubu.vv v0, v2, v4, v0.t

vwsubu.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwsubu.vv v2, v2, v4

vwadd.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwadd.vv v0, v2, v4, v0.t

vwadd.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwadd.vv v2, v2, v4

vwsub.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
0, v2, v4, v0.t

vector register group cannot overlap the source vector register group.
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR-LABEL: vwsub.vv v2, v2, v4

vwmul.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmul.vv v0, v2, v4, v0.t

vwmul.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmul.vv v2, v2, v4

vwmulu.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmulu.vv v0, v2, v4, v0.t

vwmulu.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmulu.vv v2, v2, v4

vwmulsu.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmulsu.vv v0, v2, v4, v0.t

vwmulsu.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmulsu.vv v2, v2, v4

vwmaccu.vv v0, v4, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmaccu.vv v0, v4, v2, v0.t

vwmaccu.vv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmaccu.vv v2, v4, v2

vwmacc.vv v0, v4, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmacc.vv v0, v4, v2, v0.t

vwmacc.vv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmacc.vv v2, v4, v2

vwmaccsu.vv v0, v4, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmaccsu.vv v0, v4, v2, v0.t

vwmaccsu.vv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmaccsu.vv v2, v4, v2

vfwadd.vv v0, v2, v4, v0.t
vector register group cannot overlap the mask register.
v0, v2, v4, v0.t


# CHECK-ERROR: The destination vector register group cannot overlap the source vector registe
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
r group.
# CHECK-ERROR-LABEL: vfwadd.vv v2, v2, v4

vfwsub.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwsub.vv v0, v2, v4, v0.t

vfwsub.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwsub.vv v2, v2, v4

vfwmul.vv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwmul.vv v0, v2, v4, v0.t

vfwmul.vv v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwmul.vv v2, v2, v4

vfwmacc.vv v0, v4, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwmacc.vv v0, v4, v2, v0.t

vfwmacc.vv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwmacc.vv v2, v4, v2

vfwnmacc.vv v0, v4, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwnmacc.vv v0, v4, v2, v0.t

vfwnmacc.vv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwnmacc.vv v2, v4, v2

vfwmsac.vv v0, v4, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwmsac.vv v0, v4, v2, v0.t

vfwmsac.vv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwmsac.vv v2, v4, v2

vfwnmsac.vv v0, v4, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwnmsac.vv v0, v4, v2, v0.t

vfwnmsac.vv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwnmsac.vv v2, v4, v2


vector register group cannot overlap the mask register.
v0, v2, a0, v0.t

vwaddu.vx v2, v2, a0
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwaddu.vx v2, v2, a0

vwsubu.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwsubu.vx v0, v2, a0, v0.t

vwsubu.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwsubu.vx v2, v2, a0

vwadd.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwadd.vx v0, v2, a0, v0.t

vwadd.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwadd.vx v2, v2, a0

vwsub.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwsub.vx v0, v2, a0, v0.t

vwsub.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwsub.vx v2, v2, a0

vwmul.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmul.vx v0, v2, a0, v0.t

vwmul.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmul.vx v2, v2, a0

vwmulu.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmulu.vx v0, v2, a0, v0.t

vwmulu.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmulu.vx v2, v2, a0

vwmulsu.vx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmulsu.vx v0, v2, a0, v0.t

vwmulsu.vx v2, v2, a0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmulsu.vx v2, v2, a0


vector register group cannot overlap the mask register.
v0, a0, v2, v0.t

```

```
vwmaccu.vx v2, a0, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmaccu.vx v2, a0, v2

vwmacc.vx v0, a0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmacc.vx v0, a0, v2, v0.t

vwmacc.vx v2, a0, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmacc.vx v2, a0, v2

vwmaccsu.vx v0, a0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmaccsu.vx v0, a0, v2, v0.t

vwmaccsu.vx v2, a0, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmaccsu.vx v2, a0, v2

vwmaccus.vx v0, a0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwmaccus.vx v0, a0, v2, v0.t

vwmaccus.vx v2, a0, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwmaccus.vx v2, a0, v2

vfwadd.vf v0, v2, fa0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwadd.vf v0, v2, fa0, v0.t

vfwadd.vf v2, v2, fa0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwadd.vf v2, v2, fa0

vfwsub.vf v0, v2, fa0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwsub.vf v0, v2, fa0, v0.t

vfwsub.vf v2, v2, fa0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwsub.vf v2, v2, fa0

vfwmul.vf v0, v2, fa0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwmul.vf v0, v2, fa0, v0.t

vfwmul.vf v2, v2, fa0
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
v2, v2, fa0


vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwmacc.vf v0, fa0, v2, v0.t
```

```

vfwmacc.vf v2, fa0, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwmacc.vf v2, fa0, v2

vfwnmacc.vf v0, fa0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwnmacc.vf v0, fa0, v2, v0.t

vfwnmacc.vf v2, fa0, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwnmacc.vf v2, fa0, v2

vfwmsac.vf v0, fa0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwmsac.vf v0, fa0, v2, v0.t

vfwmsac.vf v2, fa0, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwmsac.vf v2, fa0, v2

vfwnmsac.vf v0, fa0, v2, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwnmsac.vf v0, fa0, v2, v0.t

vfwnmsac.vf v2, fa0, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwnmsac.vf v2, fa0, v2

vcompress.vm v2, v2, v4
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vcompress.vm v2, v2, v4

vwaddu.wv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwaddu.wv v0, v2, v4, v0.t

vwaddu.wv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwaddu.wv v2, v4, v2

vwsubu.wv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwsubu.wv v0, v2, v4, v0.t

vwsubu.wv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwsubu.wv v2, v4, v2

vwadd.wv v0, v2, v4, v0.t
vector register group cannot overlap the mask register.
0, v2, v4, v0.t
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: The destination vector register group cannot overlap the source vector registe
```

```
r group.
# CHECK-ERROR-LABEL: vwadd.wv v2, v4, v2

vwsub.wv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwsub.wv v0, v2, v4, v0.t

vwsub.wv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vwsub.wv v2, v4, v2

vfwadd.wv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwadd.wv v0, v2, v4, v0.t

vfwadd.wv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwadd.wv v2, v4, v2

vfwsub.wv v0, v2, v4, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwsub.wv v0, v2, v4, v0.t

vfwsub.wv v2, v4, v2
# CHECK-ERROR: The destination vector register group cannot overlap the source vector register group.
# CHECK-ERROR-LABEL: vfwsub.wv v2, v4, v2

vwaddu.wx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwaddu.wx v0, v2, a0, v0.t

vwsubu.wx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwsubu.wx v0, v2, a0, v0.t

vwadd.wx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwadd.wx v0, v2, a0, v0.t

vwsub.wx v0, v2, a0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vwsub.wx v0, v2, a0, v0.t

vfwadd.wf v0, v2, fa0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwadd.wf v0, v2, fa0, v0.t

vfwsub.wf v0, v2, fa0, v0.t
# CHECK-ERROR: The destination vector register group cannot overlap the mask register.
# CHECK-ERROR-LABEL: vfwsub.wf v0, v2, fa0, v0.t
```

llvm/test/MC/RISCV/rvv/load.s

≡ View Options

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
-show-encoding --mattr=+experimental-v < %s \
# RUN:          | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vlb.v v8, (a0), v0.t
# CHECK-INST: vlb.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x04,0x05,0x10]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 05 10 <unknown>

vlb.v v8, (a0)
# CHECK-INST: vlb.v v8, (a0)
# CHECK-ENCODING: [0x07,0x04,0x05,0x12]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 05 12 <unknown>

vlh.v v8, (a0), v0.t
# CHECK-INST: vlh.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x54,0x05,0x10]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 05 10 <unknown>

vlh.v v8, (a0)
# CHECK-INST: vlh.v v8, (a0)
# CHECK-ENCODING: [0x07,0x54,0x05,0x12]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 05 12 <unknown>

vlw.v v8, (a0), v0.t
# CHECK-INST: vlw.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x64,0x05,0x10]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 05 10 <unknown>

vlw.v v8, (a0)
# CHECK-INST: vlw.v v8, (a0)
# CHECK-ENCODING: [0x07,0x64,0x05,0x12]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 05 12 <unknown>

vlbu.v v8, (a0), v0.t
# CHECK-INST: vlbu.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x04,0x05,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 05 00 <unknown>

vlbu.v v8, (a0)
# CHECK-INST: vlbu.v v8, (a0)
# CHECK-ENCODING: [0x07,0x04,0x05,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 05 02 <unknown>


                                   v0.t
                                   05,0x00]
                                   ires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 05 00 <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```

vlhu.v v8, (a0)
# CHECK-INST: vlhu.v v8, (a0)
# CHECK-ENCODING: [0x07,0x54,0x05,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 05 02 <unknown>

vlwu.v v8, (a0), v0.t
# CHECK-INST: vlwu.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x64,0x05,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 05 00 <unknown>

vlwu.v v8, (a0)
# CHECK-INST: vlwu.v v8, (a0)
# CHECK-ENCODING: [0x07,0x64,0x05,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 05 02 <unknown>

vlbff.v v8, (a0), v0.t
# CHECK-INST: vlbff.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x04,0x05,0x11]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 05 11 <unknown>

vlbff.v v8, (a0)
# CHECK-INST: vlbff.v v8, (a0)
# CHECK-ENCODING: [0x07,0x04,0x05,0x13]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 05 13 <unknown>

vlhff.v v8, (a0), v0.t
# CHECK-INST: vlhff.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x54,0x05,0x11]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 05 11 <unknown>

vlhff.v v8, (a0)
# CHECK-INST: vlhff.v v8, (a0)
# CHECK-ENCODING: [0x07,0x54,0x05,0x13]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 05 13 <unknown>

vlwff.v v8, (a0), v0.t
# CHECK-INST: vlwff.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x64,0x05,0x11]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 05 11 <unknown>

vlwff.v v8, (a0)
# CHECK-INST: vlwff.v v8, (a0)
# CHECK-ENCODING: [0x07,0x64,0x05,0x13]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 05 13 <unknown>

vlbuff.v v8, (a0), v0.t
, v0.t
05,0x01]
ires the following: 'V' (Vector Instructions)
nknown>

```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
vlbuff.v v8, (a0)
# CHECK-INST: vlbuff.v v8, (a0)
# CHECK-ENCODING: [0x07,0x04,0x05,0x03]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 05 03 <unknown>

vlhuff.v v8, (a0), v0.t
# CHECK-INST: vlhuff.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x54,0x05,0x01]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 05 01 <unknown>

vlhuff.v v8, (a0)
# CHECK-INST: vlhuff.v v8, (a0)
# CHECK-ENCODING: [0x07,0x54,0x05,0x03]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 05 03 <unknown>

vlwuff.v v8, (a0), v0.t
# CHECK-INST: vlwuff.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x64,0x05,0x01]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 05 01 <unknown>

vlwuff.v v8, (a0)
# CHECK-INST: vlwuff.v v8, (a0)
# CHECK-ENCODING: [0x07,0x64,0x05,0x03]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 05 03 <unknown>

vleff.v v8, (a0), v0.t
# CHECK-INST: vleff.v v8, (a0), v0.t
# CHECK-ENCODING: [0x07,0x74,0x05,0x01]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 74 05 01 <unknown>

vleff.v v8, (a0)
# CHECK-INST: vleff.v v8, (a0)
# CHECK-ENCODING: [0x07,0x74,0x05,0x03]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 74 05 03 <unknown>

vlsb.v v8, (a0), a1, v0.t
# CHECK-INST: vlsb.v v8, (a0), a1, v0.t
# CHECK-ENCODING: [0x07,0x04,0xb5,0x18]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 b5 18 <unknown>

vlsb.v v8, (a0), a1
# CHECK-INST: vlsb.v v8, (a0), a1
# CHECK-ENCODING: [0x07,0x04,0xb5,0x1a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 b5 1a <unknown>

vlsh.v v8, (a0), a1, v0.t
# CHECK-INST: vlsh.v v8, (a0), a1, v0.t
                                      b5,0x18]
                                      ires the following: 'V' (Vector Instructions)
                                      nknown>

vlsh.v v8, (a0), a1
```

```
# CHECK-INST: vlsh.v v8, (a0), a1
# CHECK-ENCODING: [0x07,0x54,0xb5,0x1a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 b5 1a <unknown>

vlsw.v v8, (a0), a1, v0.t
# CHECK-INST: vlsw.v v8, (a0), a1, v0.t
# CHECK-ENCODING: [0x07,0x64,0xb5,0x18]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 b5 18 <unknown>

vlsw.v v8, (a0), a1
# CHECK-INST: vlsw.v v8, (a0), a1
# CHECK-ENCODING: [0x07,0x64,0xb5,0x1a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 b5 1a <unknown>

vlsbu.v v8, (a0), a1, v0.t
# CHECK-INST: vlsbu.v v8, (a0), a1, v0.t
# CHECK-ENCODING: [0x07,0x04,0xb5,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 b5 08 <unknown>

vlsbu.v v8, (a0), a1
# CHECK-INST: vlsbu.v v8, (a0), a1
# CHECK-ENCODING: [0x07,0x04,0xb5,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 b5 0a <unknown>

vlshu.v v8, (a0), a1, v0.t
# CHECK-INST: vlshu.v v8, (a0), a1, v0.t
# CHECK-ENCODING: [0x07,0x54,0xb5,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 b5 08 <unknown>

vlshu.v v8, (a0), a1
# CHECK-INST: vlshu.v v8, (a0), a1
# CHECK-ENCODING: [0x07,0x54,0xb5,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 b5 0a <unknown>

vlswu.v v8, (a0), a1, v0.t
# CHECK-INST: vlswu.v v8, (a0), a1, v0.t
# CHECK-ENCODING: [0x07,0x64,0xb5,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 b5 08 <unknown>

vlswu.v v8, (a0), a1
# CHECK-INST: vlswu.v v8, (a0), a1
# CHECK-ENCODING: [0x07,0x64,0xb5,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 b5 0a <unknown>

vlse.v v8, (a0), a1, v0.t
# CHECK-INST: vlse.v v8, (a0), a1, v0.t
# CHECK-ENCODING: [0x07,0x74,0xb5,0x08]
                                   ires the following: 'V' (Vector Instructions)
                                   nknown>



# CHECK-INST: vlse.v v8, (a0), a1
# CHECK-ENCODING: [0x07,0x74,0xb5,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 74 b5 0a <unknown>

vlxb.v v8, (a0), v4, v0.t
# CHECK-INST: vlxb.v v8, (a0), v4, v0.t
# CHECK-ENCODING: [0x07,0x04,0x45,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 45 1c <unknown>

vlxb.v v8, (a0), v4
# CHECK-INST: vlxb.v v8, (a0), v4
# CHECK-ENCODING: [0x07,0x04,0x45,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 45 1e <unknown>

vlxh.v v8, (a0), v4, v0.t
# CHECK-INST: vlxh.v v8, (a0), v4, v0.t
# CHECK-ENCODING: [0x07,0x54,0x45,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 45 1c <unknown>

vlxh.v v8, (a0), v4
# CHECK-INST: vlxh.v v8, (a0), v4
# CHECK-ENCODING: [0x07,0x54,0x45,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 45 1e <unknown>

vlxw.v v8, (a0), v4, v0.t
# CHECK-INST: vlxw.v v8, (a0), v4, v0.t
# CHECK-ENCODING: [0x07,0x64,0x45,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 45 1c <unknown>

vlxw.v v8, (a0), v4
# CHECK-INST: vlxw.v v8, (a0), v4
# CHECK-ENCODING: [0x07,0x64,0x45,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 45 1e <unknown>

vlxbu.v v8, (a0), v4, v0.t
# CHECK-INST: vlxbu.v v8, (a0), v4, v0.t
# CHECK-ENCODING: [0x07,0x04,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 45 0c <unknown>

vlxbu.v v8, (a0), v4
# CHECK-INST: vlxbu.v v8, (a0), v4
# CHECK-ENCODING: [0x07,0x04,0x45,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 04 45 0e <unknown>

vlxhu.v v8, (a0), v4, v0.t
# CHECK-INST: vlxhu.v v8, (a0), v4, v0.t
# CHECK-ENCODING: [0x07,0x54,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
                                           nknown>


                                             v4
# CHECK-ENCODING: [0x07,0x54,0x45,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 54 45 0e <unknown>

vlxwu.v v8, (a0), v4, v0.t
# CHECK-INST: vlxwu.v v8, (a0), v4, v0.t
# CHECK-ENCODING: [0x07,0x64,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 45 0c <unknown>

vlxwu.v v8, (a0), v4
# CHECK-INST: vlxwu.v v8, (a0), v4
# CHECK-ENCODING: [0x07,0x64,0x45,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 64 45 0e <unknown>

vlxe.v v8, (a0), v4, v0.t
# CHECK-INST: vlxe.v v8, (a0), v4, v0.t
# CHECK-ENCODING: [0x07,0x74,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 74 45 0c <unknown>

vlxe.v v8, (a0), v4
# CHECK-INST: vlxe.v v8, (a0), v4
# CHECK-ENCODING: [0x07,0x74,0x45,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 74 45 0e <unknown>

vl1r.v v8, (a0)
# CHECK-INST: vl1r.v v8, (a0)
# CHECK-ENCODING: [0x07,0x74,0x85,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 07 74 85 02 <unknown>
```

llvm/test/MC/RISCV/rvv/macc.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
```

⏮ **fpallares** Not Done

Looks like `+v` attribute gets passed twice. I'm also not sure we need `+a` and `+c` .

```
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vmacc.vv v8, v20, v4, v0.t
# CHECK-INST: vmacc.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xb4]
ires the following: 'V' (Vector Instructions)
nknown>

# CHECK-INST: vmacc.vv v8, v20, v4
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ENCODING: [0x57,0x24,0x4a,0xb6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a b6 <unknown>

vmacc.vx v8, a0, v4, v0.t
# CHECK-INST: vmacc.vx v8, a0, v4, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xb4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 b4 <unknown>

vmacc.vx v8, a0, v4
# CHECK-INST: vmacc.vx v8, a0, v4
# CHECK-ENCODING: [0x57,0x64,0x45,0xb6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 b6 <unknown>

vnmsac.vv v8, v20, v4, v0.t
# CHECK-INST: vnmsac.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xbc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a bc <unknown>

vnmsac.vv v8, v20, v4
# CHECK-INST: vnmsac.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x24,0x4a,0xbe]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a be <unknown>

vnmsac.vx v8, a0, v4, v0.t
# CHECK-INST: vnmsac.vx v8, a0, v4, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xbc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 bc <unknown>

vnmsac.vx v8, a0, v4
# CHECK-INST: vnmsac.vx v8, a0, v4
# CHECK-ENCODING: [0x57,0x64,0x45,0xbe]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 be <unknown>

vmadd.vv v8, v20, v4, v0.t
# CHECK-INST: vmadd.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xa4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a a4 <unknown>

vmadd.vv v8, v20, v4
# CHECK-INST: vmadd.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x24,0x4a,0xa6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a a6 <unknown>

vmadd.vx v8, a0, v4, v0.t
# CHECK-INST: vmadd.vx v8, a0, v4, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xa4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 a4 <unknown>

                           v4
# CHECK-ENCODING: [0x57,0x64,0x45,0xa6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 a6 <unknown>

vnmsub.vv v8, v20, v4, v0.t
# CHECK-INST: vnmsub.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xac]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a ac <unknown>

vnmsub.vv v8, v20, v4
# CHECK-INST: vnmsub.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x24,0x4a,0xae]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a ae <unknown>

vnmsub.vx v8, a0, v4, v0.t
# CHECK-INST: vnmsub.vx v8, a0, v4, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xac]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 ac <unknown>

vnmsub.vx v8, a0, v4
# CHECK-INST: vnmsub.vx v8, a0, v4
# CHECK-ENCODING: [0x57,0x64,0x45,0xae]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 ae <unknown>

vwmaccu.vv v8, v20, v4, v0.t
# CHECK-INST: vwmaccu.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xf0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a f0 <unknown>

vwmaccu.vv v8, v20, v4
# CHECK-INST: vwmaccu.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x24,0x4a,0xf2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a f2 <unknown>

vwmaccu.vx v8, a0, v4, v0.t
# CHECK-INST: vwmaccu.vx v8, a0, v4, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xf0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 f0 <unknown>

vwmaccu.vx v8, a0, v4
# CHECK-INST: vwmaccu.vx v8, a0, v4
# CHECK-ENCODING: [0x57,0x64,0x45,0xf2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 f2 <unknown>

vwmacc.vv v8, v20, v4, v0.t
# CHECK-INST: vwmacc.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xf4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a f4 <unknown>


, v4
4a,0xf6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a f6 <unknown>

vwmacc.vx v8, a0, v4, v0.t
# CHECK-INST: vwmacc.vx v8, a0, v4, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xf4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 f4 <unknown>

vwmacc.vx v8, a0, v4
# CHECK-INST: vwmacc.vx v8, a0, v4
# CHECK-ENCODING: [0x57,0x64,0x45,0xf6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 f6 <unknown>

vwmaccsu.vv v8, v20, v4, v0.t
# CHECK-INST: vwmaccsu.vv v8, v20, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xfc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a fc <unknown>

vwmaccsu.vv v8, v20, v4
# CHECK-INST: vwmaccsu.vv v8, v20, v4
# CHECK-ENCODING: [0x57,0x24,0x4a,0xfe]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a fe <unknown>

vwmaccsu.vx v8, a0, v4, v0.t
# CHECK-INST: vwmaccsu.vx v8, a0, v4, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xfc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 fc <unknown>

vwmaccsu.vx v8, a0, v4
# CHECK-INST: vwmaccsu.vx v8, a0, v4
# CHECK-ENCODING: [0x57,0x64,0x45,0xfe]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 fe <unknown>

vwmaccus.vx v8, a0, v4, v0.t
# CHECK-INST: vwmaccus.vx v8, a0, v4, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xf8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 f8 <unknown>

vwmaccus.vx v8, a0, v4
# CHECK-INST: vwmaccus.vx v8, a0, v4
# CHECK-ENCODING: [0x57,0x64,0x45,0xfa]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 fa <unknown>
```

llvm/test/MC/RISCV/rvv/mask.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
                              check-prefixes=CHECK-ENCODING,CHECK-INST
                              v64 -show-encoding < %s 2>&1 \
                              check-prefix=CHECK-ERROR
                              -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vmand.mm v8, v4, v20
# CHECK-INST: vmand.mm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x66]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 66 <unknown>

vmnand.mm v8, v4, v20
# CHECK-INST: vmnand.mm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x76]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 76 <unknown>

vmandnot.mm v8, v4, v20
# CHECK-INST: vmandnot.mm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x62]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 62 <unknown>

vmxor.mm v8, v4, v20
# CHECK-INST: vmxor.mm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x6e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 6e <unknown>

vmor.mm v8, v4, v20
# CHECK-INST: vmor.mm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x6a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 6a <unknown>

vmnor.mm v8, v4, v20
# CHECK-INST: vmnor.mm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x7a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 7a <unknown>

vmornot.mm v8, v4, v20
# CHECK-INST: vmornot.mm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x72]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 72 <unknown>

vmxnor.mm v8, v4, v20
# CHECK-INST: vmxnor.mm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x7e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 7e <unknown>

vpopc.m a2, v4, v0.t
# CHECK-INST: vpopc.m a2, v4, v0.t
# CHECK-ENCODING: [0x57,0x26,0x48,0x40]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 26 48 40 <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ENCODING: [0x57,0x26,0x48,0x42]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 26 48 42 <unknown>

vfirst.m a2, v4, v0.t
# CHECK-INST: vfirst.m a2, v4, v0.t
# CHECK-ENCODING: [0x57,0xa6,0x48,0x40]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 a6 48 40 <unknown>

vfirst.m a2, v4
# CHECK-INST: vfirst.m a2, v4
# CHECK-ENCODING: [0x57,0xa6,0x48,0x42]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 a6 48 42 <unknown>

vmsbf.m v8, v4, v0.t
# CHECK-INST: vmsbf.m v8, v4, v0.t
# CHECK-ENCODING: [0x57,0xa4,0x40,0x50]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 a4 40 50 <unknown>

vmsbf.m v8, v4
# CHECK-INST: vmsbf.m v8, v4
# CHECK-ENCODING: [0x57,0xa4,0x40,0x52]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 a4 40 52 <unknown>

vmsif.m v8, v4, v0.t
# CHECK-INST: vmsif.m v8, v4, v0.t
# CHECK-ENCODING: [0x57,0xa4,0x41,0x50]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 a4 41 50 <unknown>

vmsif.m v8, v4
# CHECK-INST: vmsif.m v8, v4
# CHECK-ENCODING: [0x57,0xa4,0x41,0x52]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 a4 41 52 <unknown>

vmsof.m v8, v4, v0.t
# CHECK-INST: vmsof.m v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x41,0x50]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 41 50 <unknown>

vmsof.m v8, v4
# CHECK-INST: vmsof.m v8, v4
# CHECK-ENCODING: [0x57,0x24,0x41,0x52]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 41 52 <unknown>

viota.m v8, v4, v0.t
# CHECK-INST: viota.m v8, v4, v0.t
# CHECK-ENCODING: [0x57,0x24,0x48,0x50]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 48 50 <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
48,0x52]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 48 52 <unknown>

vid.v v8, v0.t
# CHECK-INST: vid.v v8, v0.t
# CHECK-ENCODING: [0x57,0xa4,0x08,0x50]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 a4 08 50 <unknown>

vid.v v8
# CHECK-INST: vid.v v8
# CHECK-ENCODING: [0x57,0xa4,0x08,0x52]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 a4 08 52 <unknown>
```

llvm/test/MC/RISCV/rvv/minmax.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vminu.vv v8, v4, v20, v0.t
# CHECK-INST: vminu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x10]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 10 <unknown>

vminu.vv v8, v4, v20
# CHECK-INST: vminu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x12]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 12 <unknown>

vminu.vx v8, v4, a0, v0.t
# CHECK-INST: vminu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x10]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 10 <unknown>

vminu.vx v8, v4, a0
# CHECK-INST: vminu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x12]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 12 <unknown>

vmin.vv v8, v4, v20, v0.t
# CHECK-INST: vmin.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x14]
                         ires the following: 'V' (Vector Instructions)
                         nknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vmin.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x16]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 16 <unknown>

vmin.vx v8, v4, a0, v0.t
# CHECK-INST: vmin.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x14]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 14 <unknown>

vmin.vx v8, v4, a0
# CHECK-INST: vmin.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x16]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 16 <unknown>

vmaxu.vv v8, v4, v20, v0.t
# CHECK-INST: vmaxu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x18]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 18 <unknown>

vmaxu.vv v8, v4, v20
# CHECK-INST: vmaxu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x1a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 1a <unknown>

vmaxu.vx v8, v4, a0, v0.t
# CHECK-INST: vmaxu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x18]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 18 <unknown>

vmaxu.vx v8, v4, a0
# CHECK-INST: vmaxu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x1a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 1a <unknown>

vmax.vv v8, v4, v20, v0.t
# CHECK-INST: vmax.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 1c <unknown>

vmax.vv v8, v4, v20
# CHECK-INST: vmax.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 1e <unknown>

vmax.vx v8, v4, a0, v0.t
# CHECK-INST: vmax.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 1[illegible]nknown>
[illegible]
[illegible]
[illegible]0
# CHECK-ENCODING: [0x57,0x44,0x45,0x1e]
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 1e <unknown>
```

llvm/test/MC/RISCV/rvv/mul.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vmul.vv v8, v4, v20, v0.t
# CHECK-INST: vmul.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x94]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 94 <unknown>

vmul.vv v8, v4, v20
# CHECK-INST: vmul.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x96]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 96 <unknown>

vmul.vx v8, v4, a0, v0.t
# CHECK-INST: vmul.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x94]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 94 <unknown>

vmul.vx v8, v4, a0
# CHECK-INST: vmul.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x96]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 96 <unknown>

vmulh.vv v8, v4, v20, v0.t
# CHECK-INST: vmulh.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x9c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 9c <unknown>

vmulh.vv v8, v4, v20
# CHECK-INST: vmulh.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x9e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 9e <unknown>

vmulh.vx v8, v4, a0, v0.t
# CHECK-INST: vmulh.vx v8, v4, a0, v0.t
                                  45,0x9c]
                                  ires the following: 'V' (Vector Instructions)
                                  nknown>

vmulh.vx v8, v4, a0
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vmulh.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x9e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 9e <unknown>

vmulhu.vv v8, v4, v20, v0.t
# CHECK-INST: vmulhu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x90]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 90 <unknown>

vmulhu.vv v8, v4, v20
# CHECK-INST: vmulhu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x92]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 92 <unknown>

vmulhu.vx v8, v4, a0, v0.t
# CHECK-INST: vmulhu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x90]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 90 <unknown>

vmulhu.vx v8, v4, a0
# CHECK-INST: vmulhu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x92]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 92 <unknown>

vmulhsu.vv v8, v4, v20, v0.t
# CHECK-INST: vmulhsu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x98]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 98 <unknown>

vmulhsu.vv v8, v4, v20
# CHECK-INST: vmulhsu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x9a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 9a <unknown>

vmulhsu.vx v8, v4, a0, v0.t
# CHECK-INST: vmulhsu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x98]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 98 <unknown>

vmulhsu.vx v8, v4, a0
# CHECK-INST: vmulhsu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x9a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 9a <unknown>

vwmul.vv v8, v4, v20, v0.t
# CHECK-INST: vwmul.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xec]
ires the following: 'V' (Vector Instructions)
nknown>

# CHECK-INST: vwmul.vv v8, v4, v20
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ENCODING: [0x57,0x24,0x4a,0xee]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a ee <unknown>

vwmul.vx v8, v4, a0, v0.t
# CHECK-INST: vwmul.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xec]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 ec <unknown>

vwmul.vx v8, v4, a0
# CHECK-INST: vwmul.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xee]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 ee <unknown>

vwmulu.vv v8, v4, v20, v0.t
# CHECK-INST: vwmulu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xe0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a e0 <unknown>

vwmulu.vv v8, v4, v20
# CHECK-INST: vwmulu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xe2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a e2 <unknown>

vwmulu.vx v8, v4, a0, v0.t
# CHECK-INST: vwmulu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xe0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 e0 <unknown>

vwmulu.vx v8, v4, a0
# CHECK-INST: vwmulu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xe2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 e2 <unknown>

vwmulsu.vv v8, v4, v20, v0.t
# CHECK-INST: vwmulsu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xe8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a e8 <unknown>

vwmulsu.vv v8, v4, v20
# CHECK-INST: vwmulsu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xea]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a ea <unknown>

vwmulsu.vx v8, v4, a0, v0.t
# CHECK-INST: vwmulsu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xe8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
                                     nknown>


                                        , a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xea]
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 ea <unknown>

vsmul.vv v8, v4, v20, v0.t
# CHECK-INST: vsmul.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x9c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 9c <unknown>

vsmul.vv v8, v4, v20
# CHECK-INST: vsmul.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x9e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 9e <unknown>

vsmul.vx v8, v4, a0, v0.t
# CHECK-INST: vsmul.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x9c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 9c <unknown>

vsmul.vx v8, v4, a0
# CHECK-INST: vsmul.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x9e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 9e <unknown>
```

llvm/test/MC/RISCV/rvv/mv.s

View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vmv.v.v v8, v20
# CHECK-INST: vmv.v.v v8, v20
# CHECK-ENCODING: [0x57,0x04,0x0a,0x5e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 0a 5e <unknown>

vmv.v.x v8, a0
# CHECK-INST: vmv.v.x v8, a0
# CHECK-ENCODING: [0x57,0x44,0x05,0x5e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 05 5e <unknown>

vmv.v.i v8, 15
# CHECK-INST: vmv.v.i v8, 15
                                          07,0x5e]
                                          ires the following: 'V' (Vector Instructions)
                                          nknown>

vmv.x.s a2, v4
```

```
# CHECK-INST: vmv.x.s a2, v4
# CHECK-ENCODING: [0x57,0x26,0x40,0x42]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 26 40 42 <unknown>

vmv.s.x v8, a0
# CHECK-INST: vmv.s.x v8, a0
# CHECK-ENCODING: [0x57,0x64,0x05,0x42]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 05 42 <unknown>

vmv1r.v v8, v4
# CHECK-INST: vmv1r.v v8, v4
# CHECK-ENCODING: [0x57,0x34,0x40,0x9e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 34 40 9e <unknown>

vmv2r.v v8, v4
# CHECK-INST: vmv2r.v v8, v4
# CHECK-ENCODING: [0x57,0xb4,0x40,0x9e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 40 9e <unknown>

vmv4r.v v8, v4
# CHECK-INST: vmv4r.v v8, v4
# CHECK-ENCODING: [0x57,0xb4,0x41,0x9e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 41 9e <unknown>

vmv8r.v v8, v24
# CHECK-INST: vmv8r.v v8, v24
# CHECK-ENCODING: [0x57,0xb4,0x83,0x9f]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 83 9f <unknown>
```

llvm/test/MC/RISCV/rvv/or.s

View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vor.vv v8, v4, v20, v0.t
# CHECK-INST: vor.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x28]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 28 <unknown>


                                           0
                                           4a,0x2a]
                                           ires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 2a <unknown>
```

```

vor.vx v8, v4, a0, v0.t
# CHECK-INST: vor.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x28]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 28 <unknown>

vor.vx v8, v4, a0
# CHECK-INST: vor.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x2a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 2a <unknown>

vor.vi v8, v4, 15, v0.t
# CHECK-INST: vor.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x28]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 28 <unknown>

vor.vi v8, v4, 15
# CHECK-INST: vor.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x2a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 2a <unknown>
```

llvm/test/MC/RISCV/rvv/others.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vmerge.vvm v8, v4, v20, v0
# CHECK-INST: vmerge.vvm v8, v4, v20, v0
# CHECK-ENCODING: [0x57,0x04,0x4a,0x5c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 5c <unknown>

vmerge.vxm v8, v4, a0, v0
# CHECK-INST: vmerge.vxm v8, v4, a0, v0
# CHECK-ENCODING: [0x57,0x44,0x45,0x5c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 5c <unknown>

vmerge.vim v8, v4, 15, v0
# CHECK-INST: vmerge.vim v8, v4, 15, v0
# CHECK-ENCODING: [0x57,0xb4,0x47,0x5c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 5c <unknown>



                         4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x38]
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 38 <unknown>

vslideup.vx v8, v4, a0
# CHECK-INST: vslideup.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x3a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 3a <unknown>

vslideup.vi v8, v4, 31, v0.t
# CHECK-INST: vslideup.vi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0x38]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f 38 <unknown>

vslideup.vi v8, v4, 31
# CHECK-INST: vslideup.vi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0x3a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f 3a <unknown>

vslidedown.vx v8, v4, a0, v0.t
# CHECK-INST: vslidedown.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x3c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 3c <unknown>

vslidedown.vx v8, v4, a0
# CHECK-INST: vslidedown.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x3e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 3e <unknown>

vslidedown.vi v8, v4, 31, v0.t
# CHECK-INST: vslidedown.vi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0x3c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f 3c <unknown>

vslidedown.vi v8, v4, 31
# CHECK-INST: vslidedown.vi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0x3e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f 3e <unknown>

vslide1up.vx v8, v4, a0, v0.t
# CHECK-INST: vslide1up.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x38]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 38 <unknown>

vslide1up.vx v8, v4, a0
# CHECK-INST: vslide1up.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x3a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 3a <unknown>


, v4, a0, v0.t
45,0x3c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-UNKNOWN: 57 64 45 3c <unknown>

vslide1down.vx v8, v4, a0
# CHECK-INST: vslide1down.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x3e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 3e <unknown>

vrgather.vv v8, v4, v20, v0.t
# CHECK-INST: vrgather.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x30]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 30 <unknown>

vrgather.vv v8, v4, v20
# CHECK-INST: vrgather.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x32]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 32 <unknown>

vrgather.vx v8, v4, a0, v0.t
# CHECK-INST: vrgather.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x30]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 30 <unknown>

vrgather.vx v8, v4, a0
# CHECK-INST: vrgather.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x32]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 32 <unknown>

vrgather.vi v8, v4, 31, v0.t
# CHECK-INST: vrgather.vi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0x30]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f 30 <unknown>

vrgather.vi v8, v4, 31
# CHECK-INST: vrgather.vi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0x32]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f 32 <unknown>

vcompress.vm v8, v4, v20
# CHECK-INST: vcompress.vm v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x5e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 5e <unknown>
```

llvm/test/MC/RISCV/rvv/reduction.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
                                check-prefixes=CHECK-ENCODING,CHECK-INST
                                v64 -show-encoding < %s 2>&1 \
                                check-prefix=CHECK-ERROR
                                -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vredsum.vs v8, v4, v20, v0.t
# CHECK-INST: vredsum.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 00 <unknown>

vredsum.vs v8, v4, v20
# CHECK-INST: vredsum.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 02 <unknown>

vredmaxu.vs v8, v4, v20, v0.t
# CHECK-INST: vredmaxu.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x18]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 18 <unknown>

vredmaxu.vs v8, v4, v20
# CHECK-INST: vredmaxu.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x1a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 1a <unknown>

vredmax.vs v8, v4, v20, v0.t
# CHECK-INST: vredmax.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 1c <unknown>

vredmax.vs v8, v4, v20
# CHECK-INST: vredmax.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 1e <unknown>

vredminu.vs v8, v4, v20, v0.t
# CHECK-INST: vredminu.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x10]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 10 <unknown>

vredminu.vs v8, v4, v20
# CHECK-INST: vredminu.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x12]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 12 <unknown>

vredmin.vs v8, v4, v20, v0.t
# CHECK-INST: vredmin.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x14]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
                              nknown>


                              , v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x16]
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 16 <unknown>

vredand.vs v8, v4, v20, v0.t
# CHECK-INST: vredand.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x04]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 04 <unknown>

vredand.vs v8, v4, v20
# CHECK-INST: vredand.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x06]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 06 <unknown>

vredor.vs v8, v4, v20, v0.t
# CHECK-INST: vredor.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 08 <unknown>

vredor.vs v8, v4, v20
# CHECK-INST: vredor.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 0a <unknown>

vredxor.vs v8, v4, v20, v0.t
# CHECK-INST: vredxor.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 0c <unknown>

vredxor.vs v8, v4, v20
# CHECK-INST: vredxor.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 0e <unknown>

vwredsumu.vs v8, v4, v20, v0.t
# CHECK-INST: vwredsumu.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xc0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a c0 <unknown>

vwredsumu.vs v8, v4, v20
# CHECK-INST: vwredsumu.vs v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0xc2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a c2 <unknown>

vwredsum.vs v8, v4, v20, v0.t
# CHECK-INST: vwredsum.vs v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xc4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a c4 <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
4, v20
4a,0xc6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a c6 <unknown>
```

llvm/test/MC/RISCV/rvv/shift.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vsll.vv v8, v4, v20, v0.t
# CHECK-INST: vsll.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x94]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 94 <unknown>

vsll.vv v8, v4, v20
# CHECK-INST: vsll.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x96]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 96 <unknown>

vsll.vx v8, v4, a0, v0.t
# CHECK-INST: vsll.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x94]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 94 <unknown>

vsll.vx v8, v4, a0
# CHECK-INST: vsll.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x96]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 96 <unknown>

vsll.vi v8, v4, 31, v0.t
# CHECK-INST: vsll.vi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0x94]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f 94 <unknown>

vsll.vi v8, v4, 31
# CHECK-INST: vsll.vi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0x96]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f 96 <unknown>

vsrl.vv v8, v4, v20, v0.t
# CHECK-INST: vsrl.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xa0]
                                  ires the following: 'V' (Vector Instructions)
                                  nknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vsrl.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0xa2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a a2 <unknown>

vsrl.vx v8, v4, a0, v0.t
# CHECK-INST: vsrl.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0xa0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 a0 <unknown>

vsrl.vx v8, v4, a0
# CHECK-INST: vsrl.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0xa2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 a2 <unknown>

vsrl.vi v8, v4, 31, v0.t
# CHECK-INST: vsrl.vi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xa0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f a0 <unknown>

vsrl.vi v8, v4, 31
# CHECK-INST: vsrl.vi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xa2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f a2 <unknown>

vsra.vv v8, v4, v20, v0.t
# CHECK-INST: vsra.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xa4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a a4 <unknown>

vsra.vv v8, v4, v20
# CHECK-INST: vsra.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0xa6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a a6 <unknown>

vsra.vx v8, v4, a0, v0.t
# CHECK-INST: vsra.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0xa4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 a4 <unknown>

vsra.vx v8, v4, a0
# CHECK-INST: vsra.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0xa6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 a6 <unknown>

vsra.vi v8, v4, 31, v0.t
# CHECK-INST: vsra.vi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xa4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
                                        nknown>


                                        1
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xa6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f a6 <unknown>

vnsrl.wv v8, v4, v20, v0.t
# CHECK-INST: vnsrl.wv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xb0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a b0 <unknown>

vnsrl.wv v8, v4, v20
# CHECK-INST: vnsrl.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0xb2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a b2 <unknown>

vnsrl.wx v8, v4, a0, v0.t
# CHECK-INST: vnsrl.wx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0xb0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 b0 <unknown>

vnsrl.wx v8, v4, a0
# CHECK-INST: vnsrl.wx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0xb2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 b2 <unknown>

vnsrl.wi v8, v4, 31, v0.t
# CHECK-INST: vnsrl.wi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xb0]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f b0 <unknown>

vnsrl.wi v8, v4, 31
# CHECK-INST: vnsrl.wi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xb2]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f b2 <unknown>

vnsra.wv v8, v4, v20, v0.t
# CHECK-INST: vnsra.wv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xb4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a b4 <unknown>

vnsra.wv v8, v4, v20
# CHECK-INST: vnsra.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0xb6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a b6 <unknown>

vnsra.wx v8, v4, a0, v0.t
# CHECK-INST: vnsra.wx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0xb4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 b4 <unknown>


                                 a0
                                 45,0xb6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-UNKNOWN: 57 44 45 b6 <unknown>

vnsra.wi v8, v4, 31, v0.t
# CHECK-INST: vnsra.wi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xb4]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f b4 <unknown>

vnsra.wi v8, v4, 31
# CHECK-INST: vnsra.wi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xb6]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f b6 <unknown>

vssrl.vv v8, v4, v20, v0.t
# CHECK-INST: vssrl.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xa8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a a8 <unknown>

vssrl.vv v8, v4, v20
# CHECK-INST: vssrl.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0xaa]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a aa <unknown>

vssrl.vx v8, v4, a0, v0.t
# CHECK-INST: vssrl.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0xa8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 a8 <unknown>

vssrl.vx v8, v4, a0
# CHECK-INST: vssrl.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0xaa]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 aa <unknown>

vssrl.vi v8, v4, 31, v0.t
# CHECK-INST: vssrl.vi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xa8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f a8 <unknown>

vssrl.vi v8, v4, 31
# CHECK-INST: vssrl.vi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xaa]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f aa <unknown>

vssra.vv v8, v4, v20, v0.t
# CHECK-INST: vssra.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0xac]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a ac <unknown>


v20
4a,0xae]
ires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a ae <unknown>
```

```

vssra.vx v8, v4, a0, v0.t
# CHECK-INST: vssra.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0xac]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 ac <unknown>

vssra.vx v8, v4, a0
# CHECK-INST: vssra.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0xae]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 ae <unknown>

vssra.vi v8, v4, 31, v0.t
# CHECK-INST: vssra.vi v8, v4, 31, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xac]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f ac <unknown>

vssra.vi v8, v4, 31
# CHECK-INST: vssra.vi v8, v4, 31
# CHECK-ENCODING: [0x57,0xb4,0x4f,0xae]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 4f ae <unknown>
```

llvm/test/MC/RISCV/rvv/sign-injection.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v --mattr=+f - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v --mattr=+f < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vfsgnj.vv v8, v4, v20, v0.t
# CHECK-INST: vfsgnj.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x20]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 20 <unknown>

vfsgnj.vv v8, v4, v20
# CHECK-INST: vfsgnj.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x22]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V'
(Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 22 <unknown>

vfsgnj.vf v8, v4, fa0, v0.t
# CHECK-INST: vfsgnj.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x20]
                                    ires the following: 'F' (Single-Precision Floating-Point), 'V'

                                    nknown>

vfsgnj.vf v8, v4, fa0
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vfsgnj.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x22]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 22 <unknown>

vfsgnjn.vv v8, v4, v20, v0.t
# CHECK-INST: vfsgnjn.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x24]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 24 <unknown>

vfsgnjn.vv v8, v4, v20
# CHECK-INST: vfsgnjn.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x26]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 26 <unknown>

vfsgnjn.vf v8, v4, fa0, v0.t
# CHECK-INST: vfsgnjn.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x24]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 24 <unknown>

vfsgnjn.vf v8, v4, fa0
# CHECK-INST: vfsgnjn.vf v8, v4, fa0
# CHECK-ENCODING: [0x57,0x54,0x45,0x26]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 26 <unknown>

vfsgnjx.vv v8, v4, v20, v0.t
# CHECK-INST: vfsgnjx.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x14,0x4a,0x28]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 28 <unknown>

vfsgnjx.vv v8, v4, v20
# CHECK-INST: vfsgnjx.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x14,0x4a,0x2a]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 14 4a 2a <unknown>

vfsgnjx.vf v8, v4, fa0, v0.t
# CHECK-INST: vfsgnjx.vf v8, v4, fa0, v0.t
# CHECK-ENCODING: [0x57,0x54,0x45,0x28]
# CHECK-ERROR: instruction requires the following: 'F' (Single-Precision Floating-Point), 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 54 45 28 <unknown>

vfsgnjx.vf v8, v4, fa0
# CHECK-INST: vfsgnjx.vf v8, v4, fa0
45,0x2a]
ires the following: 'F' (Single-Precision Floating-Point), 'V'

nknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

llvm/test/MC/RISCV/rvv/snippet.s

☰ View Options

This file was added.

```
## A snippet from https://github.com/riscv/riscv-v-spec.

# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN: | llvm-objdump -d --mattr=+experimental-v - \
# RUN: | FileCheck %s --check-prefix=CHECK-INST

loop:
    vsetvli a3, a0, e16,m4  # vtype = 16-bit integer vectors
# CHECK-INST: d7 76 65 00    vsetvli a3, a0, e16,m4
    vlh.v v4, (a1)          # Get 16b vector
# CHECK-INST: 07 d2 05 12    vlh.v    v4, (a1)
    slli t1, a3, 1          # Multiply length by two bytes/element
# CHECK-INST: 13 93 16 00    slli    t1, a3, 1
    add a1, a1, t1          # Bump pointer
# CHECK-INST: b3 85 65 00    add     a1, a1, t1
    vwmul.vx v8, v4, x10    # 32b in <v8--v15>
# CHECK-INST: 57 64 45 ee    vwmul.vx        v8, v4, a0

    vsetvli x0, a0, e32,m8  # Operate on 32b values
# CHECK-INST: 57 70 b5 00    vsetvli zero, a0, e32,m8
    vsrl.vi v8, v8, 3
# CHECK-INST: 57 b4 81 a2    vsrl.vi v8, v8, 3
    vsw.v v8, (a2)          # Store vector of 32b
# CHECK-INST: 27 64 06 02    vsw.v    v8, (a2)
    slli t1, a3, 2          # Multiply length by four bytes/element
# CHECK-INST: 13 93 26 00    slli    t1, a3, 2
    add a2, a2, t1          # Bump pointer
# CHECK-INST: 33 06 66 00    add     a2, a2, t1
    sub a0, a0, a3          # Decrement count
# CHECK-INST: 33 05 d5 40    sub     a0, a0, a3
    bnez a0, loop           # Any more?
# CHECK-INST: e3 1a 05 fc    bnez    a0, -44
```

llvm/test/MC/RISCV/rvv/store.s

☰ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:        | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:        | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d --mattr=+experimental-v - \
# RUN:        | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:        | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vsb.v v24, (a0), v0.t
# CHECK-INST: vsb.v v24, (a0), v0.t
                                   05,0x00]
                                   ires the following: 'V' (Vector Instructions)
                                   nknown>

vsb.v v24, (a0)
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vsb.v v24, (a0)
# CHECK-ENCODING: [0x27,0x0c,0x05,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 0c 05 02 <unknown>

vsh.v v24, (a0), v0.t
# CHECK-INST: vsh.v v24, (a0), v0.t
# CHECK-ENCODING: [0x27,0x5c,0x05,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 5c 05 00 <unknown>

vsh.v v24, (a0)
# CHECK-INST: vsh.v v24, (a0)
# CHECK-ENCODING: [0x27,0x5c,0x05,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 5c 05 02 <unknown>

vsw.v v24, (a0), v0.t
# CHECK-INST: vsw.v v24, (a0), v0.t
# CHECK-ENCODING: [0x27,0x6c,0x05,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 6c 05 00 <unknown>

vsw.v v24, (a0)
# CHECK-INST: vsw.v v24, (a0)
# CHECK-ENCODING: [0x27,0x6c,0x05,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 6c 05 02 <unknown>

vse.v v24, (a0), v0.t
# CHECK-INST: vse.v v24, (a0), v0.t
# CHECK-ENCODING: [0x27,0x7c,0x05,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c 05 00 <unknown>

vse.v v24, (a0)
# CHECK-INST: vse.v v24, (a0)
# CHECK-ENCODING: [0x27,0x7c,0x05,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c 05 02 <unknown>

vssb.v v24, (a0), a1, v0.t
# CHECK-INST: vssb.v v24, (a0), a1, v0.t
# CHECK-ENCODING: [0x27,0x0c,0xb5,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 0c b5 08 <unknown>

vssb.v v24, (a0), a1
# CHECK-INST: vssb.v v24, (a0), a1
# CHECK-ENCODING: [0x27,0x0c,0xb5,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 0c b5 0a <unknown>

vssh.v v24, (a0), a1, v0.t
# CHECK-INST: vssh.v v24, (a0), a1, v0.t
# CHECK-ENCODING: [0x27,0x5c,0xb5,0x08]
                                ires the following: 'V' (Vector Instructions)
                                nknown>


# CHECK-INST: vssh.v v24, (a0), a1
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-ENCODING: [0x27,0x5c,0xb5,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 5c b5 0a <unknown>

vssw.v v24, (a0), a1, v0.t
# CHECK-INST: vssw.v v24, (a0), a1, v0.t
# CHECK-ENCODING: [0x27,0x6c,0xb5,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 6c b5 08 <unknown>

vssw.v v24, (a0), a1
# CHECK-INST: vssw.v v24, (a0), a1
# CHECK-ENCODING: [0x27,0x6c,0xb5,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 6c b5 0a <unknown>

vsse.v v24, (a0), a1, v0.t
# CHECK-INST: vsse.v v24, (a0), a1, v0.t
# CHECK-ENCODING: [0x27,0x7c,0xb5,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c b5 08 <unknown>

vsse.v v24, (a0), a1
# CHECK-INST: vsse.v v24, (a0), a1
# CHECK-ENCODING: [0x27,0x7c,0xb5,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c b5 0a <unknown>

vsxb.v v24, (a0), v4, v0.t
# CHECK-INST: vsxb.v v24, (a0), v4, v0.t
# CHECK-ENCODING: [0x27,0x0c,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 0c 45 0c <unknown>

vsxb.v v24, (a0), v4
# CHECK-INST: vsxb.v v24, (a0), v4
# CHECK-ENCODING: [0x27,0x0c,0x45,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 0c 45 0e <unknown>

vsxh.v v24, (a0), v4, v0.t
# CHECK-INST: vsxh.v v24, (a0), v4, v0.t
# CHECK-ENCODING: [0x27,0x5c,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 5c 45 0c <unknown>

vsxh.v v24, (a0), v4
# CHECK-INST: vsxh.v v24, (a0), v4
# CHECK-ENCODING: [0x27,0x5c,0x45,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 5c 45 0e <unknown>

vsxw.v v24, (a0), v4, v0.t
# CHECK-INST: vsxw.v v24, (a0), v4, v0.t
# CHECK-ENCODING: [0x27,0x6c,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
                                nknown>


                                      v4
# CHECK-ENCODING: [0x27,0x6c,0x45,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 6c 45 0e <unknown>

vsxe.v v24, (a0), v4, v0.t
# CHECK-INST: vsxe.v v24, (a0), v4, v0.t
# CHECK-ENCODING: [0x27,0x7c,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c 45 0c <unknown>

vsxe.v v24, (a0), v4
# CHECK-INST: vsxe.v v24, (a0), v4
# CHECK-ENCODING: [0x27,0x7c,0x45,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c 45 0e <unknown>

vsuxb.v v24, (a0), v4, v0.t
# CHECK-INST: vsuxb.v v24, (a0), v4, v0.t
# CHECK-ENCODING: [0x27,0x0c,0x45,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 0c 45 1c <unknown>

vsuxb.v v24, (a0), v4
# CHECK-INST: vsuxb.v v24, (a0), v4
# CHECK-ENCODING: [0x27,0x0c,0x45,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 0c 45 1e <unknown>

vsuxh.v v24, (a0), v4, v0.t
# CHECK-INST: vsuxh.v v24, (a0), v4, v0.t
# CHECK-ENCODING: [0x27,0x5c,0x45,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 5c 45 1c <unknown>

vsuxh.v v24, (a0), v4
# CHECK-INST: vsuxh.v v24, (a0), v4
# CHECK-ENCODING: [0x27,0x5c,0x45,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 5c 45 1e <unknown>

vsuxw.v v24, (a0), v4, v0.t
# CHECK-INST: vsuxw.v v24, (a0), v4, v0.t
# CHECK-ENCODING: [0x27,0x6c,0x45,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 6c 45 1c <unknown>

vsuxw.v v24, (a0), v4
# CHECK-INST: vsuxw.v v24, (a0), v4
# CHECK-ENCODING: [0x27,0x6c,0x45,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 6c 45 1e <unknown>

vsuxe.v v24, (a0), v4, v0.t
# CHECK-INST: vsuxe.v v24, (a0), v4, v0.t
# CHECK-ENCODING: [0x27,0x7c,0x45,0x1c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c 45 1c <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
, v4
45,0x1e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c 45 1e <unknown>

vs1r.v v24, (a0)
# CHECK-INST: vs1r.v v24, (a0)
# CHECK-ENCODING: [0x27,0x7c,0x85,0x02]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 27 7c 85 02 <unknown>
```

llvm/test/MC/RISCV/rvv/sub.s

≡ View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:         | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:         | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d --mattr=+experimental-v - \
# RUN:         | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:         | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vsub.vv v8, v4, v20, v0.t
# CHECK-INST: vsub.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 08 <unknown>

vsub.vv v8, v4, v20
# CHECK-INST: vsub.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 0a <unknown>

vsub.vx v8, v4, a0, v0.t
# CHECK-INST: vsub.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x08]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 08 <unknown>

vsub.vx v8, v4, a0
# CHECK-INST: vsub.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x0a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 0a <unknown>

vrsub.vx v8, v4, a0, v0.t
# CHECK-INST: vrsub.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 0c <unknown>

vrsub.vx v8, v4, a0
# CHECK-INST: vrsub.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x0e]
                                       ires the following: 'V' (Vector Instructions)
                                       nknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-INST: vrsub.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x0c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 0c <unknown>

vrsub.vi v8, v4, 15
# CHECK-INST: vrsub.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x0e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 0e <unknown>

vwsubu.vv v8, v4, v20, v0.t
# CHECK-INST: vwsubu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xc8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a c8 <unknown>

vwsubu.vv v8, v4, v20
# CHECK-INST: vwsubu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xca]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a ca <unknown>

vwsubu.vx v8, v4, a0, v0.t
# CHECK-INST: vwsubu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xc8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 c8 <unknown>

vwsubu.vx v8, v4, a0
# CHECK-INST: vwsubu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xca]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 ca <unknown>

vwsub.vv v8, v4, v20, v0.t
# CHECK-INST: vwsub.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xcc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a cc <unknown>

vwsub.vv v8, v4, v20
# CHECK-INST: vwsub.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xce]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a ce <unknown>

vwsub.vx v8, v4, a0, v0.t
# CHECK-INST: vwsub.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xcc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 cc <unknown>

vwsub.vx v8, v4, a0
# CHECK-INST: vwsub.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xce]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 nknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xd8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a d8 <unknown>

vwsubu.wv v8, v4, v20
# CHECK-INST: vwsubu.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xda]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a da <unknown>

vwsubu.wx v8, v4, a0, v0.t
# CHECK-INST: vwsubu.wx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xd8]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 d8 <unknown>

vwsubu.wx v8, v4, a0
# CHECK-INST: vwsubu.wx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xda]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 da <unknown>

vwsub.wv v8, v4, v20, v0.t
# CHECK-INST: vwsub.wv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0xdc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a dc <unknown>

vwsub.wv v8, v4, v20
# CHECK-INST: vwsub.wv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0xde]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a de <unknown>

vwsub.wx v8, v4, a0, v0.t
# CHECK-INST: vwsub.wx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0xdc]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 dc <unknown>

vwsub.wx v8, v4, a0
# CHECK-INST: vwsub.wx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0xde]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 de <unknown>

vsbc.vvm v8, v4, v20, v0
# CHECK-INST: vsbc.vvm v8, v4, v20, v0
# CHECK-ENCODING: [0x57,0x04,0x4a,0x48]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 48 <unknown>

vsbc.vxm v8, v4, a0, v0
# CHECK-INST: vsbc.vxm v8, v4, a0, v0
# CHECK-ENCODING: [0x57,0x44,0x45,0x48]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 48 <unknown>


                                v20, v0
                               4a,0x4c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-UNKNOWN: 57 04 4a 4c <unknown>

vmsbc.vxm v8, v4, a0, v0
# CHECK-INST: vmsbc.vxm v8, v4, a0, v0
# CHECK-ENCODING: [0x57,0x44,0x45,0x4c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 4c <unknown>

vmsbc.vv v8, v4, v20
# CHECK-INST: vmsbc.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x4e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 4e <unknown>

vmsbc.vx v8, v4, a0
# CHECK-INST: vmsbc.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x4e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 4e <unknown>

vssubu.vv v8, v4, v20, v0.t
# CHECK-INST: vssubu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x88]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 88 <unknown>

vssubu.vv v8, v4, v20
# CHECK-INST: vssubu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x8a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 8a <unknown>

vssubu.vx v8, v4, a0, v0.t
# CHECK-INST: vssubu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x44,0x45,0x88]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 88 <unknown>

vssubu.vx v8, v4, a0
# CHECK-INST: vssubu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x8a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 8a <unknown>

vssub.vv v8, v4, v20, v0.t
# CHECK-INST: vssub.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x8c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 8c <unknown>

vssub.vv v8, v4, v20
# CHECK-INST: vssub.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x8e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 8e <unknown>


                                  a0, v0.t
                                  45,0x8c]
                                  ires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 8c <unknown>
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```

vssub.vx v8, v4, a0
# CHECK-INST: vssub.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x8e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 8e <unknown>

vasub.vv v8, v4, v20, v0.t
# CHECK-INST: vasub.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x2c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 2c <unknown>

vasub.vv v8, v4, v20
# CHECK-INST: vasub.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x2e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 2e <unknown>

vasub.vx v8, v4, a0, v0.t
# CHECK-INST: vasub.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x2c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 2c <unknown>

vasub.vx v8, v4, a0
# CHECK-INST: vasub.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x2e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 2e <unknown>

vasubu.vv v8, v4, v20, v0.t
# CHECK-INST: vasubu.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x24,0x4a,0x28]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 28 <unknown>

vasubu.vv v8, v4, v20
# CHECK-INST: vasubu.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x24,0x4a,0x2a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 24 4a 2a <unknown>

vasubu.vx v8, v4, a0, v0.t
# CHECK-INST: vasubu.vx v8, v4, a0, v0.t
# CHECK-ENCODING: [0x57,0x64,0x45,0x28]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 28 <unknown>

vasubu.vx v8, v4, a0
# CHECK-INST: vasubu.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x64,0x45,0x2a]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 64 45 2a <unknown>
```

llvm/test/MC/RISCV/rvv/vsetvl.s

☰ View Options

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:          | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:          | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:          | llvm-objdump -d --mattr=+experimental-v - \
# RUN:          | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:          | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

```

kito-cheng Not Done

Add a testcase for `vsetvli a2, a0, e32`, spec say LMUL can be omitted and default to m1[1], and need negative testcase would be better, e.g. feed `e31` and `m3`.

[1] https://riscv.github.io/documents/riscv-v-spec/#_vsetvlivsetvl_instructions

```
vsetvli a2, a0, e32,m4
# CHECK-INST: vsetvli a2, a0, e32,m4
# CHECK-ENCODING: [0x57,0x76,0xa5,0x00]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 76 a5 00 <unknown>

vsetvl a2, a0, a1
# CHECK-INST: vsetvl a2, a0, a1
# CHECK-ENCODING: [0x57,0x76,0xb5,0x80]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 76 b5 80 <unknown>
```

llvm/test/MC/RISCV/rvv/xor.s

View Options

This file was added.

```
# RUN: llvm-mc -triple=riscv64 -show-encoding --mattr=+experimental-v < %s \
# RUN:          | FileCheck %s --check-prefixes=CHECK-ENCODING,CHECK-INST
# RUN: not llvm-mc -triple=riscv64 -show-encoding < %s 2>&1 \
# RUN:          | FileCheck %s --check-prefix=CHECK-ERROR
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:          | llvm-objdump -d --mattr=+experimental-v - \
# RUN:          | FileCheck %s --check-prefix=CHECK-INST
# RUN: llvm-mc -triple=riscv64 -filetype=obj --mattr=+experimental-v < %s \
# RUN:          | llvm-objdump -d - | FileCheck %s --check-prefix=CHECK-UNKNOWN

vxor.vv v8, v4, v20, v0.t
# CHECK-INST: vxor.vv v8, v4, v20, v0.t
# CHECK-ENCODING: [0x57,0x04,0x4a,0x2c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 2c <unknown>

vxor.vv v8, v4, v20
# CHECK-INST: vxor.vv v8, v4, v20
# CHECK-ENCODING: [0x57,0x04,0x4a,0x2e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 04 4a 2e <unknown>

0, v0.t
45,0x2c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
```

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.

```
# CHECK-UNKNOWN: 57 44 45 2c <unknown>

vxor.vx v8, v4, a0
# CHECK-INST: vxor.vx v8, v4, a0
# CHECK-ENCODING: [0x57,0x44,0x45,0x2e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 44 45 2e <unknown>

vxor.vi v8, v4, 15, v0.t
# CHECK-INST: vxor.vi v8, v4, 15, v0.t
# CHECK-ENCODING: [0x57,0xb4,0x47,0x2c]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 2c <unknown>

vxor.vi v8, v4, 15
# CHECK-INST: vxor.vi v8, v4, 15
# CHECK-ENCODING: [0x57,0xb4,0x47,0x2e]
# CHECK-ERROR: instruction requires the following: 'V' (Vector Instructions)
# CHECK-UNKNOWN: 57 b4 47 2e <unknown>
```

Add Action...

Submit

Your browser timezone setting differs from the timezone setting in your profile, click to reconcile.