Buddy Compiler

We ❤️ MLIR Playground

Default File

Share Button

Sample File

rvv-vp-intrinsic-add.mlir ← Choose Examples

Code Input

Source MLIR Code

```
memref.global "private" @gv_i32 : memref<20xi32> = dense<[0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
                                                          10, 11, 12, 13, 14, 15, 16, 17, 18, 19]>
memref.global "private" @gv_f32 : memref<20xf32> = dense<[0. , 1. , 2. , 3. , 4. , 5. , 6. , 7. , 8. , 9. ,
                                                          10., 11., 12., 13., 14., 15., 16., 17., 18., 19.]>
func.func @main() -> i32 {
  %mem_i32 = memref.get_global @gv_i32 : memref<20xi32>
  %mem_f32 = memref.get_global @gv_f32 : memref<20xf32>
  %c0 = arith.constant 0 : index
  %c1 = arith.constant 1 : index
  %c6 = arith.constant 6 : index
  %c10 = arith.constant 10 : index
  %mask6 = arith.constant dense<[1, 1, 1, 1, 1, 0, 0]> : vector<8xi1>
  %evl8 = arith.constant 8 : i32
  %mask8 = arith.constant dense<[1, 1, 1, 1, 1, 1, 1, 1]> : vector<8xi1>
  %evl6 = arith.constant 6 : i32
  %c1_i32 = arith.constant 1 : i32
  %c1_f32 = arith.constant 1.0 : f32

  //===----------------------------------------------------------------------===//
  // VP Intrinsic Add Operation + Fixed Vector Type
  //===----------------------------------------------------------------------===//
```

Compilation Stages → Lower Box | Translate Box | Compile Box | Link Box | Execute Box

--convert-scf-to-cf | --convert-vector-to-llvm | --convert-memref-to-llvm | --convert-arith-to-llvm | --convert-func-to-llvm

Pass Pipeline

Result Output

Intermediate Product

```
module attributes {llvm.data_layout = ""} {
  llvm.func @printF32(f32)
  llvm.func @printNewline()
  llvm.func @printClose()
  llvm.func @printComma()
  llvm.func @printOpen()
  llvm.func @printI64(i64)
  llvm.mlir.global private @gv_i32(dense<[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]>
  llvm.mlir.global private @gv_f32(dense<[0.000000e+00, 1.000000e+00, 2.000000e+00, 3.000000e+00, 4.000000e+00, 
  llvm.func @main() -> i32 {
    %0 = llvm.mlir.constant(0 : i32) : i32
    %1 = llvm.mlir.constant(1.000000e+00 : f32) : f32
    %2 = llvm.mlir.constant(1 : i32) : i32
    %3 = llvm.mlir.constant(6 : i32) : i32
    %4 = llvm.mlir.constant(dense<true> : vector<8xi1>) : vector<8xi1>
    %5 = llvm.mlir.constant(8 : i32) : i32
    %6 = llvm.mlir.constant(dense<[true, true, true, true, true, true, false, false]> : vector<8xi1>) : vector<8
    %7 = llvm.mlir.constant(10 : index) : i64
    %8 = llvm.mlir.constant(0 : index) : i64
    %9 = llvm.mlir.constant(20 : index) : i64
    %10 = llvm.mlir.constant(1 : index) : i64
```

(a) Buddy-CAAS Main Interface

Config Picker

Configs

--convert-scf-to-cf --convert-vector-to-llvm --convert-memref-to-llvm --convert-arith-to-llvm --convert-func-to-llvm --reconcile-unrealized-casts

Lowering

--convert-scf-to-cf --lower-rvv --lower-bud --convert-vector-to-llvm --convert-memref-to-llvm --convert-arith-to-llvm --convert-func-to-llvm --reconcile-unrealized-casts

Search: vector

--force-32bit-vector-indices

--convert-vector-to-scf

--convert-vector-to-spirv

(b) Config-Picker tool

Lower Box | Translate Box | Compile Box | Link Box | Execute Box

--convert-scf-to-cf | --convert-vector-to-llvm | --convert-memref-to-llvm | --convert-arith-to-llvm | --convert-func-to-llvm

Result Output

```
module {
  llvm.mlir.global private @gv_i32(dense<[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]>
  llvm.mlir.global private @gv_f32(dense<[0.000000e+00, 1.000000e+00, 2.000000e+00, 3.000000e+00, 4.000000e+00, 
  func.func @main() -> i32 {
    %0 = llvm.mlir.constant(20 : index) : i64
    %1 = llvm.mlir.constant(1 : index) : i64
    %2 = llvm.mlir.null : !llvm.ptr<i32>
    %3 = llvm.getelementptr %2[%0] : (!llvm.ptr<i32>, i64) -> !llvm.ptr<i32>
```

(c) Disable some passes in the pipeline

Lower Box | Translate Box | Compile Box | Link Box | Execute Box

--convert-scf-to-cf | --convert-memref-to-llvm | --convert-vector-to-llvm | --convert-arith-to-llvm | --convert-func-to-llvm

(d) Drag and drop a pass

Lower Box | Translate Box | Compile Box | Link Box | Execute Box

-L | thirdparty/build-riscv-gnu-toolchain/sysroot | -cpu | rv64,x-v=true,vlen=128 | a.out

Log

```
( 10, 12, 14, 16, 18, 20, 6, 7 )
( 10, 12, 14, 16, 18, 20, 6, 7 )
( 10, 12, 14, 16, 18, 20, 6, 7 )
( 10, 12, 14, 16, 18, 20, 6, 7 )
```

(e) Execute on a specific backend using the emulator