中国科学院软件研究所
Institute of Software Chinese Academy of Sciences

智能软件研究中心
Intelligent Software Research Center

软件所智能软件中心PLCT实验室 王鹏 实习生

2020//02/19

目 录

• 01 c910扩展指令集添加介绍

u@u-virtual-machine: ~/tools/c910-llvm-master/tmp/llvm/lib/Target/RISCV

File Edit View Search Terminal Help

```
//
//
//===----------------------------------------------------------------------===//
// Instruction Formats
//===----------------------------------------------------------------------===//
include "RISCVInstrFormats.td"

def EXT : RVInstC910BO_1<0b010, OPC_CUSTOM0,
                         (outs GPR:$rd), (ins GPR:$rs1, simm12:$imm1, simm12:$imm2),
                         "ext", "$rd, $rs1, $imm1, $imm2">;
def EXTU : RVInstC910BO_1<0b011, OPC_CUSTOM0,
                         (outs GPR:$rd), (ins GPR:$rs1, simm12:$imm1, simm12:$imm2),
                         "extu", "$rd, $rs1, $imm1, $imm2">;
```

改为RISCVInstrFormatsC910.td

u@u-virtual-machine: ~/tools/c910-llvm-master/tmp/llvm/lib/Target/RISCV

File Edit View Search Terminal Help

```
                        "'D' (Double-Precision Floating-Point)",
                        [FeatureStdExtF]>;
def HasStdExtD : Predicate<"Subtarget->hasStdExtD()">,
                             AssemblerPredicate<"FeatureStdExtD">;

def FeatureStdExtC
    : SubtargetFeature<"c", "HasStdExtC", "true",
                        "'C' (Compressed Instructions)">;
def HasStdExtC : Predicate<"Subtarget->hasStdExtC()">,
                             AssemblerPredicate<"FeatureStdExtC">;

def FeatureC910
    : SubtargetFeature<"c910", "HasC910", "true",
                        "Implement C910">;
def HasC910 : Predicate<"Subtarget->hasC910()">,
                             AssemblerPredicate<"FeatureC910">;
```

RISCV.td中添加C910

RISCV/RISCVSubtarget.h

u@u-virtual-machine: ~/tools/c910-llvm-master/tmp/llvm/lib/Target/RISCV

File Edit View Search Terminal Help

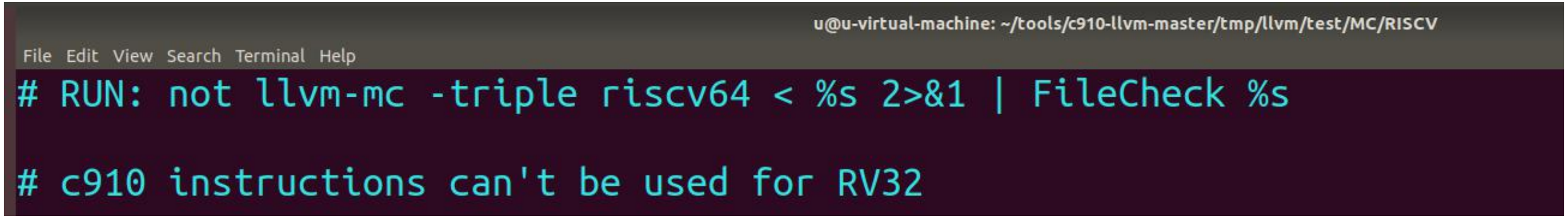
```
#define GET_SUBTARGETINFO_HEADER
#include "RISCVGenSubtargetInfo.inc"

namespace llvm {
class StringRef;

class RISCVSubtarget : public RISCVGenSubtargetInfo {
  virtual void anchor();
  bool HasStdExtM = false;
  bool HasStdExtA = false;
  bool HasStdExtF = false;
  bool HasStdExtD = false;
  bool HasStdExtC = false;
  bool HasC910 = false;
  bool HasRV64 = false;
  bool IsRV32E = false;
  bool EnableLinkerRelax = false;
  unsigned XLen = 32;
  MVT XLenVT = MVT::i32;
  RISCVABI::ABI TargetABI = RISCVABI::ABI_Unknown;
```

```
  bool hasStdExtM() const { return HasStdExtM; }
  bool hasStdExtA() const { return HasStdExtA; }
  bool hasStdExtF() const { return HasStdExtF; }
  bool hasStdExtD() const { return HasStdExtD; }
  bool hasStdExtC() const { return HasStdExtC; }
  bool hasC910() const { return HasC910; }
  bool is64Bit() const { return HasRV64; }
  bool isRV32E() const { return IsRV32E; }
  bool enableLinkerRelax() const { return EnableLinkerRelax; }
  MVT getXLenVT() const { return XLenVT; }
  unsigned getXLen() const { return XLen; }
  RISCVABI::ABI getTargetABI() const { return TargetABI; }
};
```

test/MC/RISCV/c910-valid.s



test/MC/RISCV/c910-invalid.s

```
//===----------------------------------------------------------------------===//
// RISC-V processors supported.
//===----------------------------------------------------------------------===//

def : ProcessorModel<"c910", NoSchedModel, [Feature64Bit,FeatureStdExtA,FeatureStdExtC,
                                            FeatureStdExtM,FeatureStdExtF,FeatureStdExtD,FeatureC910]>;

def : ProcessorModel<"generic-rv32", NoSchedModel, []>;

def : ProcessorModel<"generic-rv64", NoSchedModel, [Feature64Bit]>;
```

RISCV.td

重新编译成功

开始测试

```
u@u-virtual-machine:~/tools/c910-llvm-master/tmp/build$ echo "vsetvl a3, a3, a2" | llvm-mc -triple=
riscv64 -mcpu=c910 --mattr=+c910 -show-encoding -show-inst
        .text
<stdin>:1:1: error: unrecognized instruction mnemonic
vsetvl a3, a3, a2
^
```

```
u@u-virtual-machine:~/tools/c910-llvm-master/tmp/build$ echo "ff0 a0,a1" | llvm-mc --triple=riscv64
 -mcpu=c910 -mattr=+c910 -show-encoding -show-inst
        .text
        ff0     a0, a1                  # encoding: [0x0b,0x95,0x05,0x80]
                                        # <MCInst #385 FF0
                                        #  <MCOperand Reg:11>
                                        #  <MCOperand Reg:12>>
u@u-virtual-machine:~/tools/c910-llvm-master/tmp/build$ echo "ext a0,a1,4,1" | llvm-mc --triple=ris
cv64 -mcpu=c910 -mattr=+c910 -show-encoding -show-inst
        .text
        ext     a0, a1, 4, 1            # encoding: [0x0b,0xa5,0x25,0x20]
                                        # <MCInst #354 EXT
                                        #  <MCOperand Reg:11>
                                        #  <MCOperand Reg:12>
                                        #  <MCOperand Imm:4>
                                        #  <MCOperand Imm:1>>
```

```
u@u-virtual-machine:~/tools/c910-llvm-master/tmp/build$ echo "extu a0,a1,5,1" | llvm-mc --triple=ri
scv64 -mcpu=c910 -mattr=+c910 -show-encoding -show-inst
        .text
        extu    a0, a1, 5, 1            # encoding: [0x0b,0xb5,0x25,0x28]
                                        # <MCInst #355 EXTU
                                        #  <MCOperand Reg:11>
                                        #  <MCOperand Reg:12>
                                        #  <MCOperand Imm:5>
                                        #  <MCOperand Imm:1>>
u@u-virtual-machine:~/tools/c910-llvm-master/tmp/build$ echo "rev a0,a1" | llvm-mc --triple=riscv64
 -mcpu=c910 -mattr=+c910 -show-encoding -show-inst
        .text
        rev     a0, a1                  # encoding: [0x0b,0x95,0x05,0x82]
                                        # <MCInst #453 REV
                                        #  <MCOperand Reg:11>
                                        #  <MCOperand Reg:12>>
```

```
u@u-virtual-machine:~/tools/c910-llvm-master/tmp/build$ echo "tst a0,a1,2" | llvm-mc --triple=riscv
64 -mcpu=c910 -mattr=+c910 -show-encoding -show-inst
        .text
<stdin>:1:1: error: unrecognized instruction mnemonic
tst a0,a1,2
^
u@u-virtual-machine:~/tools/c910-llvm-master/tmp/build$ echo "tstnbz a0,a1" | llvm-mc --triple=risc
v64 -mcpu=c910 -mattr=+c910 -show-encoding -show-inst
        .text
        tstnbz  a0, a1                  # encoding: [0x0b,0x95,0x05,0x80]
                                        # <MCInst #487 TSTNBZ
                                        #  <MCOperand Reg:11>
                                        #  <MCOperand Reg:12>>
u@u-virtual-machine:~/tools/c910-llvm-master/tmp/build$ echo "ff1 a0,a1" | llvm-mc --triple=riscv64
 -mcpu=c910 -mattr=+c910 -show-encoding -show-inst
        .text
        ff1     a0, a1                  # encoding: [0x0b,0x95,0x05,0x80]
                                        # <MCInst #386 FF1
                                        #  <MCOperand Reg:11>
                                        #  <MCOperand Reg:12>>
```

• 参考资料

isrc-cas/rvv-llvm

https://github.com/isrc-cas/rvv-llvm/tree/rvv-iscas

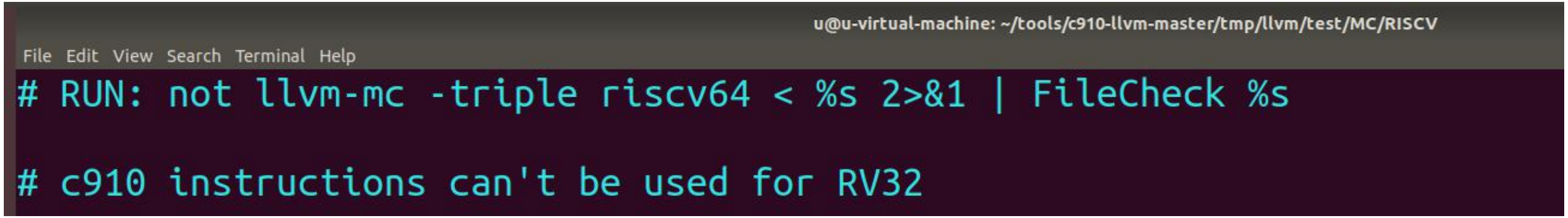
《LLVM Codebook》

Extending RISC-V ISA With a Custom Instruction Set Extension

https://www.design-reuse.com/articles/46237/extending-risc-v-isa-with-a-custom-instruction-set-extension.html

• 课后问题

test/MC/RISCV/c910-valid.s



test/MC/RISCV/c910-invalid.s

谢 谢

欢迎交流合作

2019/02/25