Supporting TVM on RISC-V Architectures

Jenq-Kuen Lee¹, Allen Lu², Yuan-Ming Chang¹,², Chao-Lin Lee¹,²
Piyo Chen¹, and Shao-Chung Wang³

¹Department of Computer Science, National Tsing Hua University, Taiwan
²Peakhills Group Corporation
³Andes Technology Corporation
RISC-V with two vector ISAs to support fall-back engine with AI Models

Super Word Vector

Packed Vector (SubWord SIMD) With Fixed-Point and Integer Instructions

Add Sub Mul Div Compare Signed Unsigned

RISC-V DSP (P) Extension Proposal Chuan-Hua Chang, Andes Technology Corporation

Courtesy: Vector ISA, Roger Espasa, Esperanto Technologies
• We add RISC-V target in TVM codegen phase. The TVM RISC-V codegen will lower SIMD computation with Subword SIMD intrinsics.
• The LLVM backend will need to generate the corresponding SIMD instructions.
• Also on-going work to add TVM scheduling to quantize computation into fixed-points, “quantize(width, exponent)”. 

expr CodeGenRISCV::CreateAddIntr(const Add::op,
::llvm::Intrinsic::ID vadd_id = ::llvm::Int

if(bits==8) {
  vadd_id=::llvm::Intrinsic::riscv_simd_sad
}
const Expr& el = op->a;
const Expr& e2 = op->b;
Array<Expr> vcnt_args;
vcnt_args.push_back(ir::UIpImm::make(UInt(32), vpaddlu_id));
vcnt_args.push_back(ir::UIpImm::make(UInt(32), 0));
vcnt_args.push_back(e1);
vcnt_args.push_back(e2);
return ir::Call::make(op->type, "llvm_intrin", vcnt_args, Call::PureIntrinsic);
Example – Matrix Multiply

In this example, 104 of 229 instructions will be with SIMD computation which process two element in one instruction.
Summary and Future Work

• Also has some discussions with AWS team to add RISC-V back-end for TVM deep learning compiler.
• Look forward to contributing the codes to TVM source trees.
• Currently the work is with Spike RISC-V simulator and we look forward to using Gem5 and Sid simulators and real chips for performance tuning.