Supporting TVM on RISC-V Architectures with SIMD Computations

Jenq-Kuen Lee[1] , Chun-Chieh Yang[1], Allen Lu[2], P. Chen[1], YM Chang[1,2], CH Chang[1], Yi-Ru Chen[1], HH Liao[1], Chao-Lin Lee[1,2], Ssu-Hsuan Lu[2] , and Shao-Chung Wang[3]

[1]Department of Computer Science, National Tsing Hua University, Taiwan
[2]Peakhills Group Corporation
[3]Andes Technology Corporation

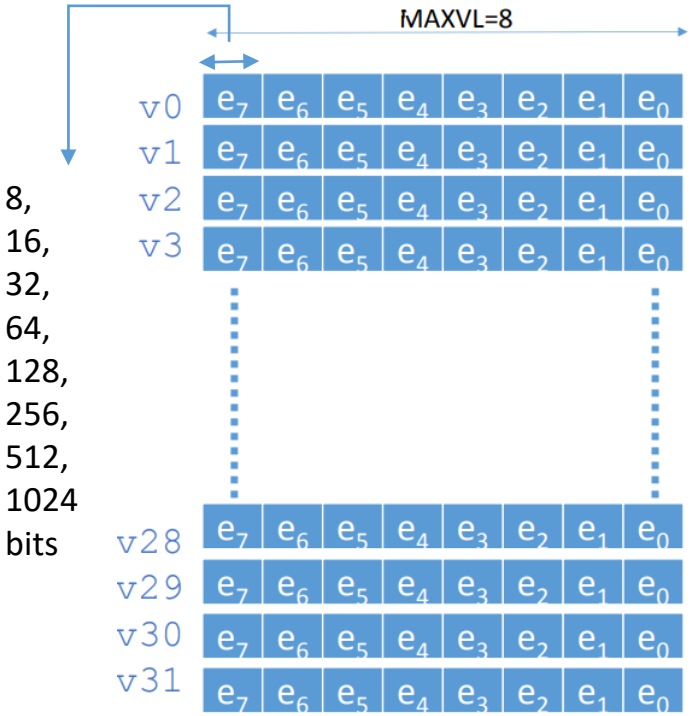
PEAKHILLS GROUP

ANDES TECHNOLOGY

PL NTHU LAB

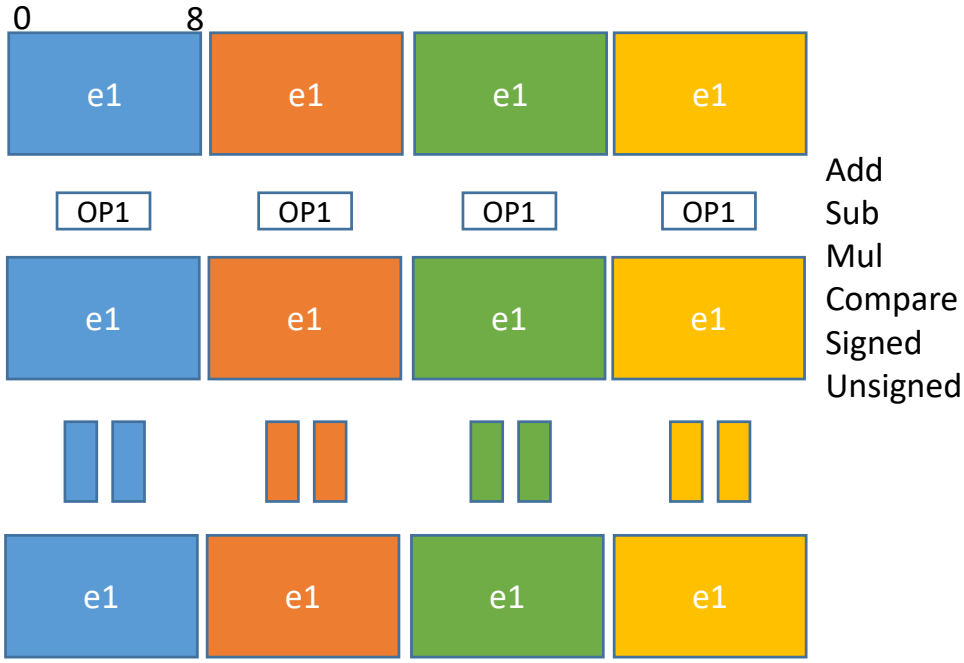
RISC-V with two vector ISAs to support fall-back engine with AI Models

Super Word Vector V Extension



Courtesy: Vector ISA, Roger Espasa, Esperanto Technologies

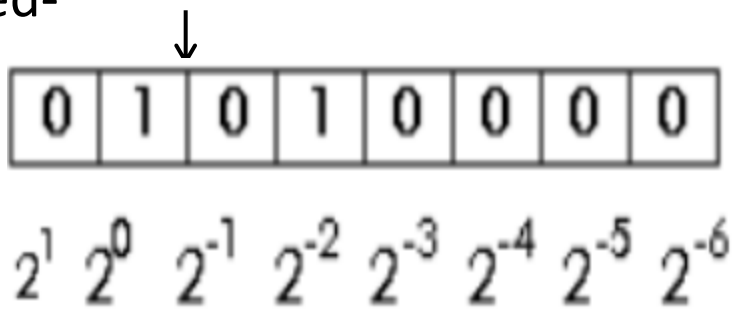
Packed Vector (SubWord SIMD) P Extension With Fixed-Point and Integer Instructions



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

[RFC] Fixed-point type implementation proposal #4446

- RISC-V P extension (Subword SIMD) with fixed-point instructions.
- We refer Fxp as fixed-point value, Fp as floating-point value and PP as point position
  - Fxp = Fp * pow(2,PP)
- Support Fixed-point Type with TVM



= 1+ ¼ = 1.25

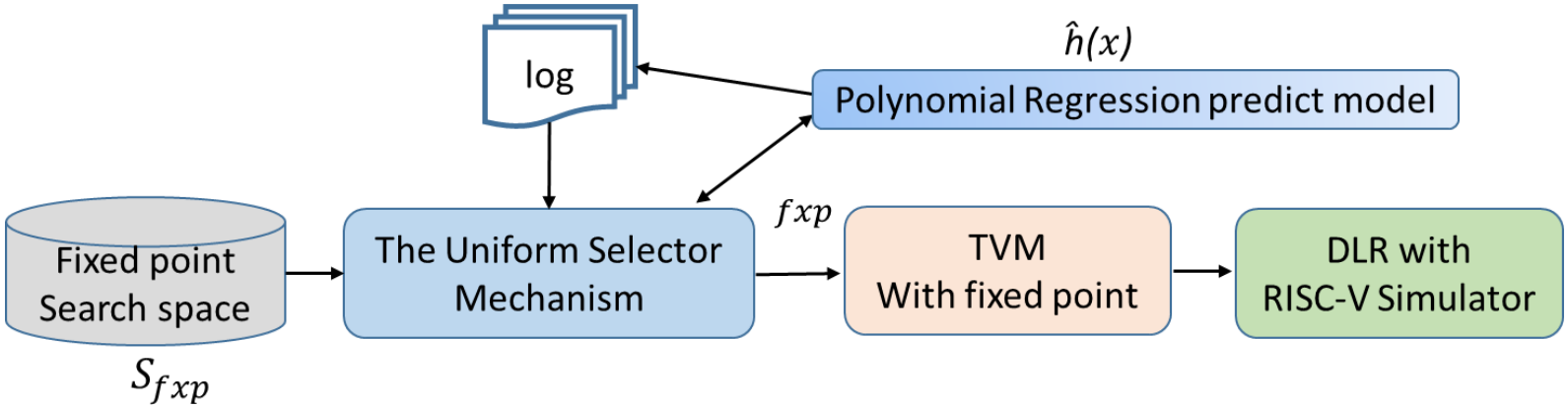
- Compiler time with type information for the binary point position of the variable.

References for Fixed-Point Type

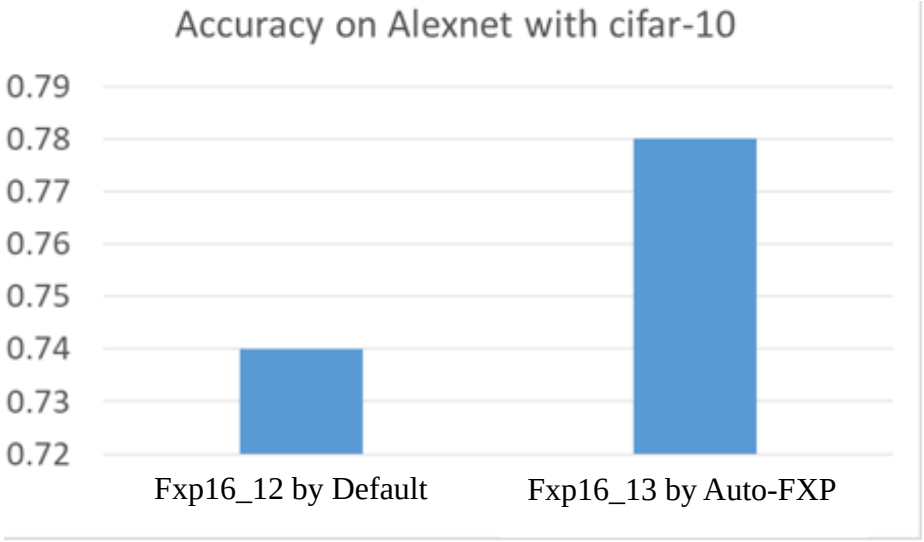
(1) AC fixed-Point by Mentor graphics (https://www.mentor.com/hls-lp/downloads/ac-datatypes)

(2) Our early proposal to Khronos for OpenCL fixed-point feature set (https://www.khronos.org/assets/uploads/developers/library/2018-khronos-group-opencl-embedded-outreach/Taipei-DSP-Profile-NTHU_Jan18.pdf)

Auto-FXP with TVM on RISC-V with p Extension



- Using machine learning model to auto-tune the binary point position.
- It can find the best binary point position for fixed-point expression when we have TVM on RISC-V with p extension.
- The work extends AutoTVM and can enhance the accuracy while enjoy the low power numeric benefits.
- The tuning work is done with spike simulator incorporated with RISC-V P extension (Subword SIMD).

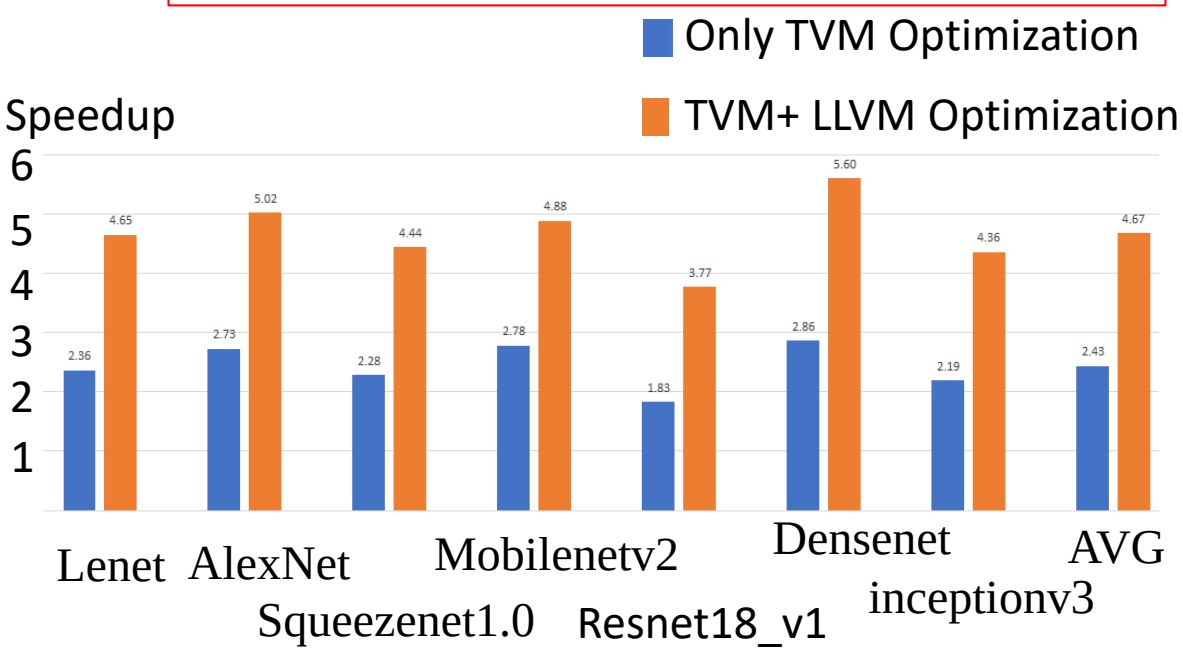


TVM for RISC-V with V Extension (Superword SIMD)

- TVM Optimization
  - The TVM RISC-V codegen will lower SIMD computation with SIMD intrinsics into LLVM.
  - The LLVM backend will need to generate the corresponding SIMD instructions.
  - Need to tune the scheduler to provide a large loop index space for vector parallelism.
- LLVM Optimization
  - VSETVL Redundancy Elimination
  - VMulADD Resource Utilization
  - Fast Vector Initializer

- Spike Simulator
- assume 512 bits vector register
- V SIMD in <4 x float32>,
- <8 x float32>, <16 x float32>
- Spec v0.7.0, TVM v0.6, LLVM 9.0.0
- Compare with SIMD float32 and no SIMD float32

Speedup based on runtime executed instructions



Summary

- Thank you AWS team help with AI model validation flow.
- Look forward to contributing codes to the TVM source trees.
- More detailed of our work can also be found in the following.
  - Experiments and AI Model Validations for Neo/TVM on RISC-V Architectures with SIMD, Allen Lu, et al, RISC-V Summit, San Jose, Dec 2019 (Poster).
  - Enabling TVM on RISC-V Architectures with SIMD Instructions, Allen Lu, Chao-Lin Lee, Yuan-Ming Chang, Piyo Chen, Hsiang-Wei Sung, Heng Lin, Shao-Chung Wang, and Jenq-Kuen Lee, RISC-V Forum, March 2019 (Oral presentation).