Libre-SOC SVP64 Vector Processing

Augmenting the OpenPOWER ISA
to provide 3D and Video instructions
and add Cray-style Vector Extensions

ICS2021

Sponsored by NLnet's PET Programme

June 14, 2021

OpenPOWER today

- Open ISA: EULA v3.0B announced August 2019
- Compliancy subsets: mandatory and optional features
- Compliance provides royalty-free IBM Patent grant
- Custom extensions permitted (see v3.0C): recommends "common-usage" ones be submitted as RFCs to OpenPOWER ISA WG
- On this basis we have the freedom and are encouraged to create Cray-style Vectorisation Extensions
- VSX will not be part of that: it is fixed-width SIMD.
  https://tinyurl.com/simd-considered-harmful
  https://en.wikipedia.org/wiki/Vector_processor

Why OpenPOWER?

- Good ecosystem essential
  linux kernel, u-boot, compilers, OSes,
  Reference Implementation(s)
- Supportive Foundation and Members
  need to be able to submit ISA augmentations
  (for proper peer review)
- No NDAs, full transparency must be acceptable
  due to being funded under NLnet's PET Programme
- OpenPOWER: established for decades, excellent Foundation,
  Microwatt as Reference, approachable and friendly.

Severe Limitations of RISC-V for Supercomputing

- Independent Research and public commentary: https://news.ycombinator.com/item?id=24459314 https://www.iscaconf.org/isca2020/papers/466100a052.pdf
- No LOAD/STORE with Update (present in OpenPOWER)
- No LOAD/STORE with Shift-immediate (as in ARM, x86)
- No Condition Codes or Carry (present in OpenPOWER) Extremely costly to add to compiler infrastructure (already done in OpenPOWER).
- Over-simplified ISA (assumption of macro-op fusion and Compressed which massively complicates Multi-issue decode and Issue phases, Multi-issue being fundamental to HPC)
- RISC-V is great for Embedded scenarios, but it is just not up to scratch for Supercomputing. OpenPOWER already is.

The summary on SVP64

- Specification: https://libre-soc.org/openpower/sv/svp64/
- SVP64 is similar to Intel x86 "REP" instruction
  "please repeat the following instruction N times"
  (but add some extra "stuff" in the process)
- Uses the Cray-style "setvl" instruction
  (Cray-I, NEC SX-Aurora, RISC-V RVV)
- Unlike "REP" there is additional "Vector context":
  Predication, Twin-predication, Element-width Overrides,
  Map-reduce, Iteration, Saturation and more.
- Just like "REP", none of this requires extra instructions!
  (except setvl and the "REP"-like prefix itself)
- "SIMD Considered Harmful" principle applies equally to
  RISC-V Vectors (190+ instructions on top of RV64GC's 80)
  RVV more than doubles the number of RISC-V instructions.

Simple-V ADD in a nutshell

```
function op_add(rd, rs1, rs2, predr) # add not VADD!
 int i, id=0, irs1=0, irs2=0;
 for (i = 0; i < VL; i++)
   if (ireg[predr] & 1<<i) # predication uses intregs
     ireg[rd+id] <= ireg[rs1+irs1] + ireg[rs2+irs2];
    if (reg_is_vectorised[rd] ) { id += 1; }
    if (reg_is_vectorised[rs1]) { irs1 += 1; }
    if (reg_is_vectorised[rs2]) { irs2 += 1; }
```

- Above is oversimplified: Reg. indirection left out (for clarity).
- SIMD slightly more complex (case above is elwidth = default)
- Scalar-scalar and scalar-vector and vector-vector now all in one
- OoO may choose to push ADDs into instr. queue (v. busy!)

Additional Simple-V features

- "fail-on-first" (POWER9 VSX strncpy segfaults on boundary!)
- "Twin Predication" (covers VSPLAT, VGATHER, VSCATTER, VINDEX etc.)
- SVP64: extensive "tag" (Vector context) augmentation
- "Context propagation": a VLIW-like context. Allows contexts to be repeatedly applied (x86 "REP"). Effectively a "hardware compression algorithm" for ISAs.
- Map-reduce and Iteration (like Cray-I and SX-Aurora). Also new: prefix-sum (Pascal's Triangle)
- REMAP (suitable for in-place variable-sized Matrix Multiply)
- Ultimate goal: cut down I-Cache usage, cuts down on power
- Specifications: https://libre-soc.org/openpower/sv/
- Needs to go through OpenPOWER Foundation 'approval'

How can you help?

- We need help. This is big. plenty of Research Opportunities
  Also a lot more "Supercomputer-centric"
- Start here! https://libre-soc.org
  Mailing lists https://lists.libre-soc.org
  IRC Freenode libre-soc
  etc. etc. (it's a Libre project, go figure)
- Can I get paid? Yes! NLnet funded
  See https://libre-soc.org/nlnet/#faq
- Also profit-sharing in any commercial ventures

Summary

- OpenPOWER is already a Supercomputer ISA: SVP64 greatly simplifies it, goes back to the original Cray-style roots, and adds half a dozen completely new innovations.
- Collaboration with OpenPOWER Foundation and Members absolutely essential. No short-cuts. Standards to be developed and ratified so that everyone benefits.
- Riding the wave of huge stability of OpenPOWER ecosystem
- Greatly simplified software development: reduces costs and risks.
- It also happens to be fascinating, deeply rewarding technically challenging, and funded by NLnet
- Simulator and HDL at https://git.libre-soc.org/

The end

Thank you

Questions?

- Discussion: http://lists.libre-soc.org
- OFTC IRC #libre-soc
- http://libre-soc.org/
- http://nlnet.nl/PET
- https://libre-soc.org/nlnet/#faq