Supporting the RISC-V Vector Extension in LLVM

Robin Kruppe, Julian Oppermann, Andreas Koch
Embedded Systems and Applications Group (ESA), TU Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

[llvm-dev] RFC: Supporting the RISC-V vector extension in LLVM

Robin Kruppe via llvm-dev llvm-dev at lists.llvm.org
Wed Apr 11 02:44:52 PDT 2018

RISC-V

Processor-dependent vector length

TECHNISCHE UNIVERSITÄT DARMSTADT

- SIMD/vectorized code, but vector length chosen by microarchitecture
- Compile software once, run on all processors
- Strip-mined loops: handle as much work per iteration as hardware supports
- see also: Arm Scalable Vector Extension (SVE)

Configuration determined vector length

TECHNISCHE UNIVERSITÄT DARMSTADT

- RISC-V vector unit is configurable: registers count, bit width of vector elements, ...
- Vector length determined from configuration!
- Configuration tailored to kernel can greatly improve perf & energy & ...
- Want compiler to generate tailored configurations, but this means:
- Vector length not just unknown (like SVE), also changes at run time

Proposed IR extension: dynamic-length vector type

TECHNISCHE UNIVERSITÄT DARMSTADT

```
%p = alloca <vlen x i32>
%v = call <vlen x i32> @llvm.riscv.foo()
store <vlen x i32> %v, <vlen x i32>* %p
```

- New first-class type (possibly reuse SVE vector type)
- Vector length mustn't change in the middle of vector code
- To keep this manageable, declare: vector length changes on calls & returns

Proposed IR extension: dynamic-length vector type with vector length token

TECHNISCHE UNIVERSITÄT DARMSTADT

```
%L = vlentoken
%p = alloca <vlen x i32>, vlen %L
%v = call <vlen x i32> @llvm.riscv.foo(), vlen %L
store <vlen x i32> %v, <vlen x i32>* %p, vlen %L
```

- Problem: some passes (e.g., outlining) move instructions between functions
- `token` type leveraged to keep vector operations together in same function
- One `vlentoken` per function to simplify IR passes

Current status

- Implemented IR changes
- Prototyped MIR support: instructions, vector unit state, configuration
- Vector unit configuration by piggy-backing on register allocator
- Next up: loop vectorization (possibly via VPlan), ISel support

Interested? Questions? Thoughts? Objections?

TECHNISCHE UNIVERSITÄT DARMSTADT

Find me at the poster session! or email me: `kruppe@esa.tu-darmstadt.de`

Discuss the full RFC on llvm-dev!

`lists.llvm.org/pipermail/llvm-dev/2018-April/122517.html`

[llvm-dev] RFC: Supporting the RISC-V vector extension in LLVM

Robin Kruppe via llvm-dev llvm-dev at lists.llvm.org
Wed Apr 11 02:44:52 PDT 2018