Porting SW to riscv64

Ludovic HENRY
About

Software Engineer & Team Lead at Rivos

- Managed Runtimes, System Libraries, Profiling

Language Runtimes WG at RISC-V

- “collaborative effort [...] to accelerate the development of open source software for the RISC-V architecture”
- OpenJDK, Go, Python, .NET, ART, V8
- Compilers, Runtimes, and Ecosystem (libraries, tools)

Adoptium WG

- Distributing LTS versions (11, 17, 21 in progress)
Intended Audience

- Some experience with RISC-V and who want to get more involved
- No experience with RISC-V but it sounds exciting (it is!)

- I will not talk Assembly
- If you don’t know a word or concept, please ask!

- Targeting “application” systems (smartphone, laptop, desktop, servers, hpc)
Thank you AArch64!

- Blueprint for changes: all necessary #ifdef, build/cross-compilation scripts, CI setup, and more
- Lots of project are either [1] x86+aarch64+ppc64+s390x or [2] x86-only
  - Rarely in-between
  - Easy to add riscv64 to [1]
  - Need a lot more work for [2]
    - Build system: teach non-x86 specificities, cross-compilation
    - Sources: stub-out x86-specifics, memory model
    - CI: let’s dive into that later
Resources

● RISC-V GitHub org: spread out but the most complete
  ○ https://github.com/riscv/riscv-isa-manual/releases - scalar instructions
  ○ https://github.com/riscv/riscv-v-spec/releases/tag/v1.0 - vector instructions
  ○ https://github.com/riscv/riscv-crypto/releases/tag/v1.0.0 - vector crypto instructions
  ○ https://github.com/riscv-non-isa/rrv-intrinsic-doc/releases/tag/v1.0-rc0 - vector intrinsics
  ○ Watch out for Pre-Releases, things may (will!) change in subtle ways

● RISC-V V Intrinsics viewer
  ○ https://dzaima.github.io/intrinsics-viewer/ (14094 results 😱)
Targets

- **Families of extensions**
  - rv64gc
  - Bitmanip: Zba, Zbb, Zbs
  - Vector: V
  - Vector Crypto: Zvbb, Zvbc, Zvkg, Zvkn, and more

- **Profiles: rva20, rva22, rva23**
  - [https://github.com/riscv/riscv-profiles/releases/tag/v1.0](https://github.com/riscv/riscv-profiles/releases/tag/v1.0)
  - Certainty: rv64gc + bitmanip + hwprobe for V and vector crypto
  - Expectations: future is rva23 + vector crypto (*)

- **hwprobe is your friend**
  - Checks for extension availability at run-time and more
Compilers / Runtimes / Libraries

● Support in many compilers/runtimes
  ○ GCC, LLVM, OpenJDK, Go, Python, .NET, V8, ART, and many more
  ○ Various degrees of quality and support
  ○ Rapidly evolving
  ○ Importance of latest and greatest

● Support in more and more libraries
  ○ Most of the upcoming work
  ○ Gotta love transitive dependencies
Compilers / Runtimes / Libraries

- [https://landscape.riscv.org](https://landscape.riscv.org)
Compilers / Runtimes / Libraries

- Huge shoutout to all the contributors!
  - Many doing it on their free time
Gotchas / Difficulties

- Here be dragons assumptions!
- Vector Length Specific
  - Different than SSE/AVX, Neon
  - Many libraries just assume vectors are fixed lengths
- Canonical NaNs
  - Different behavior than x86
  - Sign of a NaN?
- Memory model
  - Strong (x86 TSO) vs. Weak (RVWMO)
- RVV simplicity
  - Simple to program, Hard to implement in HW
  - Depends on the microarchitecture
  - Requires broad testing
Developing / Compiling / Testing

● QEMU is your friend!
  ○ Functionally: the most complete
  ○ User-space emulation is “easy” enough
    ■ `apt install qemu-user-static`
    ■ `docker run -it riscv64/ubuntu bash`
  ○ Great for (most) testing
  ○ Not perfect though
    ■ Leaky abstraction (ex: /proc/cpuinfo)
    ■ Not fast, particularly for linking large libraries/executables
    ■ Debugging gets complicated
Developing / Compiling / Testing

● Cross-compilation + Testing on dev boards
  ○ Faster build times
  ○ Today’s boards have limitations
    ■ Don’t support everything; vector, vector crypto
    ■ HW Bugs
QEMU is your friend (again)!

- 1-liner on GitHub Actions
  - uses: docker/setup-qemu-action@v3
- You don’t even need docker!
  - Create yourself a sysroot (see debootstrap)
  - Set QEMU_LD_PREFIX=/path/to/sysroot
  - And voilà
- Tweak available extensions with QEMU_CPU
  - rv64,zba=true,zbb=true,zbs=true,v=false
  - rv64,zba=true,zbb=true,zbs=true,v=true,vlen=128
  - rv64,zba=true,zbb=true,zbs=true,v=true,vlen=256
Performance Measurements

● QEMU is NOT your friend!
  ○ Not cycle accurate (vendor specific, secret)
  ○ Maybe instruction count (very inaccurate!)

● Boards
  ○ Imagine optimizing for AWS Graviton 4 by measuring on 1st gen Raspberry Pi
    ■ In-order CPU, few cores, limited scalability
  ○ Only 1 supports vector today (CanMV k230)
  ○ It’s getting better (slowly)!

● Optimization Manual
  ○ Upcoming in next few days
Closing Thoughts

It’s FUN, and never too late. So much more work than we can handle!

Check out https://wiki.riseproject.dev for SW work to be done

If you have an idea, make a proposal (paid OSS work!)

THANK YOU TO ALL CONTRIBUTORS!
Contacts

Mastodon: @ludovic_dev@mastodon.social

Blog: https://blog.ludovic.dev/

Email: mail@ludovic.dev

Any questions? Please, ping me!
Learn more

- https://github.com/openjdk/jdk
- https://github.com/golang/go
- https://github.com/shibatch/sleef
- https://github.com/xtensor-stack/xsimd
- https://github.com/openssl/openssl
- https://github.com/netty/netty