RISC-V Memory Consistency Model Status Update

Dan Lustig and the Memory Model TG
May 8, 2018
MEMORY MODEL TASK GROUP PROGRESS

• May 2017 Workshop: Formed the task group
  (...debate...)

• November 2017 Workshop: Settled on the basics
  • RVWMO baseline, and optional RVTSO extension
    (...refinement...)

• May 2018 Workshop: released for ratification!
  • Public review period runs May 2 through June 16
WHAT WERE OUR GOALS?

• Define the RISC-V *memory consistency model*
  • Specifies the values that can be returned by loads

• Support a wide range of RISC-V implementations

• Support Linux, C/C++, and lots of other critical SW
RISC-V MEMORY MODEL SPECIFICATION

- Chapter 6: RISC-V Weak Memory Ordering (“RVWMO”)
- Chapter 20: “Zam” Std. Extension for Misaligned AMOs
- Chapter 21: “Ztso” Std. Extension for Total Store Ordering
- Appendix A: Explanatory Material and Litmus Tests
- Appendix B: Formal Memory Model Specifications
RVWMO RULES IN A NUTSHELL

- **Global Memory Order**: a total order over the memory operations generated by the instructions in each program
- GMO is constrained by **Preserved Program Order**: 

```
A
Overlap
Store
except “rsw” and “fri;rfi”

Load
Overlap
Load

AMO/SC
Overlap
Load

A
Fence
B

.aq
A
.rl
B

.aq
A
.rl
B

SC
RCsc

LR

Addr/ctrl/ data dep.
“(addr|data);rfi” or “addr;po”

except ctrl deps. where B is a load
```
RVWMO RULES IN A NUTSHELL

• **Load Value Axiom:** each byte of each load $i$ returns the value written to that byte by the store that is the latest in global memory order among the following stores:
  1. Stores that write that byte and that precede $i$ in the global memory order
  2. Stores that write that byte and that precede $i$ in program order

• **Atomicity Axiom:** no store from another hart can appear in the global memory order between a paired LR and successful SC
  
  *(this axiom simplified here for clarity...see spec for complete definition)*

• **Progress Axiom:** no memory operation may be preceded in the global memory order by an infinite sequence of other memory operations
MEMORY MODEL ISA EXTENSIONS

- “Zam” extends “A” by permitting misaligned AMOs
  - “A” without “Zam” now forbids misaligned AMOs or LR/SC pairs

- “Ztso” strengthens the baseline memory model to TSO
  - TSO-only code is not backwards-compatible with RVWMO
DOCUMENTATION & TOOLS

- Appendix A: two dozen pages explaining the details in plain English
- Appendix B: Two axiomatic models and one operational model, with associated tools (Alloy, herd, rmem)
- More than 7000 litmus tests online
  - (also to be used to test compliance)

```
// Prepro
fun ppo : (* Generate global memory order *)

let gno = (* precursor: ie build gno as an total order that include gno *)
    gno & (W/V) = FW | # Final write after any write to the same location

```

Figure A.16: Because of the address dependency from (d) to (e), (d) also precedes (f) (outcome forbidden)
ONGOING/FUTURE WORK

• Mixed-size, partially-overlapping memory accesses
• Formalize instruction fetches and FENCE.I TLB flushes and SFENCE.VMA, etc.
• Integration with other extensions (V, J, N, T, ...)
• Integration with the ISA formalization task group’s effort
• Cache flush/writeback/etc. operations
• (The task group logistics for all this are still TBD)
MEMORY MODEL RATIFICATION TIMELINE

- Released for public review on 5/2/18
- Foundation requires at least 45 days for public review. This will end no earlier than 6/16/18.
- If you have comments or feedback:
  - send to isa-dev
  - send as a PR or issue on riscv-isa-manual GitHub repo
  - send to me directly