Hardware Updating of PTE A/D Bits (Svadu)

Ved Shanbhogue

Version 0.1, 9/2022: This document is in stable state. See http://riscv.org/spec-state for details.

Table of Contents

Preamble

⚠ *This document is in the Stable state*

Assume anything could still change, but limited change should be expected.

Copyright and license information

Contributors

This RISC-V specification has been contributed (in alphabetical order) to directly or indirectly by:

Aaron Durbin, Andrew Waterman, Earl Killian, Greg Favor, John Ingalls, Krste Asanović, Paul Donahue, Ved Shanbhogue

Hardware Updating of PTE A/D Bits

The Svadu extension adds support and CSR controls for hardware updating of PTE A/D bits. The A and D bits are managed by these extensions as follows:

- When a virtual page is accessed and the A bit is clear, the PTE is updated to set the A bit. When the virtual page is written and the D bit is clear, the PTE is updated to set the D bit. When G-stage address translation is in use and is not Bare, the G-stage virtual pages may be accessed or written by implicit accesses to VS-level memory management data structures, such as page tables.
- When two-stage address translation is in use, an explicit access may cause both VS-stage and G-stage PTEs to be updated. The following rules apply to all PTE updates caused by an explicit or an implicit memory accesses.

  The PTE update must be atomic with respect to other accesses to the PTE, and must atomically check that the PTE is valid and grants sufficient permissions. Updates of the A bit may be performed as a result of speculation, but updates to the D bit must be exact (i.e., not speculative), and observed in program order by the local hart. When two-stage address translation is active, updates of the D bit in G-stage PTEs may be performed as a result of speculative updates of the A bit in VS-stage PTEs.

  The PTE update must appear in the global memory order before the memory access that caused the PTE update and before any subsequent explicit memory access to that virtual page by the local hart. The ordering on loads and stores provided by FENCE instructions and the acquire/release bits on atomic instructions also orders the PTE updates associated with those loads and stores as observed by remote harts.

  The PTE update is not required to be atomic with respect to the memory access that caused the update and a trap may occur between the PTE update and the memory access that caused the PTE update. If a trap occurs then the A and/or D bit may be updated but the memory access that caused the PTE update may not occur. The hart must not perform the memory access that caused the PTE update before the PTE update is globally visible.

Svadu extension requires the page tables to be located in cacheable main memory PMA regions.

The translation of virtual addresses (or guest physical addresses) to physical (or guest physical) addresses is accomplished with the same algorithm as specified in the Supervisor-Level ISA extension (section 4.3.2) and as modified by the hypervisor extension (section 8.5.1), except that step 7 of the translation process, instead of causing a page-fault exception due to A and/or D bits being 0 when required to be 1, continues as follows:

7. If `pte.a = 0`, or if the original memory access is a store and `pte.d = 0`:
   a. If a store to `pte` would violate a PMA or PMP check, raise an access-fault exception corresponding to the original access type.
   b. Perform the following steps atomically:
      i. Compare `pte` to the value of the PTE at address `a + va.vpn[i] × PTESIZE`.

ii. If the values match, set `pte.a` to 1 and, if the original memory access is a store, also set `pte.d` to 1.

iii. If the comparison fails, return to step 2

The Svadu extension adds the `HADE` bit (bit 61) to `menvcfg`. When `menvcfg.HADE` is 1, hardware updating of PTE A/D bits is enabled during single-stage address translation. When the hypervisor extension is implemented, if `menvcfg.HADE` is 1, hardware updating of PTE A/D bits is enabled during G-stage address translation. When `menvcfg.HADE` is zero, the implementation behaves as though Svadu were not implemented. If Svadu is not implemented, `menvcfg.HADE` is read-only zero. Furthermore, for implementations with the hypervisor extension, `henvcfg.HADE` is read-only zero if `menvcfg.HADE` is zero.

When the hypervisor extension is implemented, the Svadu extension adds the `HADE` bit (bit 61) to `henvcfg`. When `henvcfg.HADE` is 1, hardware updating of PTE A/D bits is enabled during VS-stage address translation. When `henvcfg.HADE` is zero, the implementation behaves as though Svadu were not implemented.