IA32 Paging Perspective: OS Developer view

Konstantin Belousov `kib@freebsd.org`

March 18, 2017

`git src: 2017-03-17 15:37:57 +0200 58f66dd`

Agenda

Talk Content

- Introduction
- IA32 Paging evolution
- FreeBSD VM
- IA32 PCID and its use in amd64 pmap

Questions

- Short questions in line
- Discussion after the blocks and at the end of the talk

Virtual Memory

Design Choises

- Multiple or Single Address Space (MAS vs. SAS)
- Segments or Arbitrary Mappings
- Sparce or Compact AS

HW optimized for common OS design

IA32

Early 198x, 'VAX compatible'

Intel 386

- Hierarchial two-level page table, 4K pages.
- Flat and large enough (4G) per-process address space.
- Permissions: r/w, u/s. (Valid, Accessed, Dirty) per page.
- Hardware page walker.
- OS-visible TLB, no RAM or SMP coherence.

386 bugs

- No write protection from kernel, Copy-on-Write requires invalidation. Fixed in 486.

Evolution

First tweaks

TLB – Translation Lookaside Buffer

- Upper limit on memory access speed
- Must execute before any cache access (PIPT)
- Must be fully associative. Power, Transistors budget

Pentium, 1993

- Global pages: TLB items not flushed on context switch
- PSE, Superpages: use less TLB items for same AS coverage

PAE

Pentium Pro, 1996

- PAE: 36bit physical address (64G addressable), 32bit virtual (4G), 3-level page table (3rd level is fake)

AMD Athlon, 2003

- nx bit
- long mode (64 bit): 52bit physical, 48bit virtual (256 TBytes), 4-level pages.

FreeBSD VM

Descendant of Mach VM, 199x, university of Carnegie-Mellon.

Machine-Independent Layer

- Main principle: laziness.
- Abstract representation of address space elements

PMAP

HW-abstraction layer

AMD64 pmap

Mechanisms

- pv lists
- Direct MAP
- TLB coherency (SMP IPIs)
- Transparent superpages

PCID

Associate Process ID with each TLB entry

Goal: avoid flushing translations on context switch.

Implementation 1: Naive

- Static Process ID assignment
- Avoid TLB shootdown on context switch
- Send shootdown IPI to all CPUs which activated AS

Downsides

- Eventually all shootdowns become broadcast
- Too many races maintaining active CPUs for given AS

PCID

Implementation 2: SVR3/MIPS

Vahalia. Unix Internal. The New Frontiers.

- Ephemeral Process ID, per CPU
- Avoid TLB shootdown on context switch
- Send shootdown IPI to active CPUs
- Inactive CPUs re-allocate Process ID when activating

Measurements

- No context switch latency change
- But global 10x reduction of TLB misses

CPU Bugs

Intel

Spurious page faults on Nehalem (Core I7 900)

323254-024US. Errata BC93. An Unexpected Page Fault or EPT Violation May Occur After Another Logical Processor Creates a Valid Translation of a Page.

AMD

Phenom TLB Bug

Revision Guide for AMD Family 10h Processors doc. 41322 Rev. 3.92 March 2012 Errata 298. L2 Eviction May Occur During Processor Operation To Set Accessed or Dirty Bit

CPU Bugs: not only x86

Cavium ThunderX

Erratum 26026. No public documentation. Atomics might behave incorrectly if broadcast TLBI is executed on another CPU in parallel to STX.

References

- Intel 64 and IA-32 Architectures Software Developer Manuals, Volume 3
- AMD, AMD64 Architecture Programmer's Manual Volume 2: System Programming
- Vahalia. Unix Internal. The New Frontiers. P.–H., 1996
- FreeBSD commits r255060, r282684

Questions

Use The Source