ASID Management in Xen AMD-V
Partitioning the physical TLB with SVM ASIDs

Sebastian Biemueller
Xen Summit Spring 2007
Concept of Address Space IDs (ASIDs)

- Software defines current ASID
- ASID field added to each TLB tag
  - Derived from current ASID when TLB entry is loaded
  - Can participate in the TLB-hit logic
- Allows dynamic partitioning of the physical TLB
ASIDs in SVM

• Active while SVM enabled
• Hypervisor and host-level applications: ASID = 0
• VM guests: ASID = {1..63}

• VMRUN:
  – User-controlled flush of complete physical TLB
  – Switch to hypervisor-defined ASID

• #VMEXIT:
  – Does not flush the physical TLB
  – Implicit switch to ASID = 0
Analysis

• Current Status in Xen
  – Only one ASID used for all guests
  – Each VMRUN configured to flush of the physical TLB

• Observation
  – ASIDs are CPU-local resource
  – Number of ASIDs is limited -> contention
  – Recycling an ASID requires TLB-flush
  – TLB-flush automatically recycles all ASIDs
Design for Xen

- Each VCPU gets an ASID
- Guest or Hypervisor action may require TLB flush
  - Instead, simply assign new ASID
- ASIDs assigned round-robin
- ASID invalidation by versioning (TLB generation)

- Optimize for common cases
  - ASID reuse
  - Guest causes TLB flush
Resume VCPU

1. ASID valid?
   - VCPU’s TLB generation equals CPU’s TLB generation
2. New ASID required?
   - Assign free ASID to VCPU
3. Out of free ASIDs?
   - Flush physical TLB (recycles all ASIDs)
   - Generation++ (avoid explicit VCPU invalidation)
4. Generation overflow?
   - Designed not to happen
Use Cases

1. Resume current VCPU
   - Minimized overhead (2 cache hits)
   - Invalid Generation triggers ASID management

2. Resume other VCPU
   - VCPU may live in invalid generation

3. VCPU creation/migration
   - Assign invalid generation
   - Enforces ASID assignment on first/next resume
Evaluation Environment

- Test cases
  - 2x2 (32bit guest on top of 32bit hypervisor)
  - 4x4 (64bit guest on top of 64bit hypervisor)
- Hypervisor
  - Xen with changeset 14703
- Guest OS
  - SUSE10 with 2.6GB memory, UP, SMP, VCPUs pinned
- Benchmark
  - Kernbench
- Procedure
  - Benchmarks runs 3 times
  - Elapsed time of benchmark as measured using host’s TSC
Evaluation

- TLB flushes reduced to 1.5% ($\frac{1}{\text{No. of ASIDs}}$)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Occurrence</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASID check</td>
<td>100%</td>
<td>20 cycles</td>
</tr>
<tr>
<td>ASID assignment</td>
<td>30%</td>
<td>42 cycles</td>
</tr>
<tr>
<td>Increase generation</td>
<td>0.3%</td>
<td>460-600 cycles</td>
</tr>
<tr>
<td>Generation overflow</td>
<td>“never”</td>
<td></td>
</tr>
</tbody>
</table>

• Conclusion
  - Guest performance increased drastically
  - Kernbench 11% faster
Trademark Attribution

AMD, the AMD Arrow logo, AMD-V and AMD Virtualization, and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. Other names used in this presentation are for identification purposes only and may be trademarks of their respective owners.

©2007 Advanced Micro Devices, Inc. All rights reserved.