Virtual Hierarchies to Support Server Consolidation

Mike Marty
Mark Hill
University of Wisconsin-Madison- ISCA 2007

Presented for ECE259 by:
Mohammad Mottaghi – Feb. 15th 2010
Objective

Design of a memory system for many-core CMP’s to:

- Maximize shared memory accesses serviced within a VM
- Minimize interference among separate VM’s
- Facilitate dynamic reassignment of VM’s to Proc & Mem

Virtualization goals:

- functional isolation
- performance isolation
- dynamic resource reassignment
- content-based page sharing
  - VMWare: Up to 60% reduced memory due to CB page sharing
Existing techniques

1- DRAM directory
2- TAG Directory
3- Static-Bank Directory

<table>
<thead>
<tr>
<th></th>
<th>Tag DIR</th>
<th>Static Bank Dir</th>
<th>STATIC-BANK-DIR w/ hypervisor-managed cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimize Performance</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Isolate Performance</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Allow Dynamic Partitioning</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
<tr>
<td>Support Inter-VM Sharing</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Hypervisor/OS Simplicity</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
</tbody>
</table>
Virtual Hierarchies

---

Key Idea:

Overlay 2-level Coherence Hierarchy on physically-flat CMP

- First level **harmonizes** with VM/Workload
- Second level allows inter-VM sharing, migration, reconfig
  - Two strategies:
    - VH_A ➔ directory-based
    - VH_B ➔ broadcast-based
VH: First-Level Protocol

Intra-VM Directory Protocol w/ interleaved directories

Dynamic home tile selected by VM Config Table

- Hardware VM Config Table at each tile
- Set by hypervisor during scheduling

Full bit-vector to track any possible sharer

- Intra-VM broadcast also possible
- 64 rows of 6-bit vectors: lookup before a miss leaves its tile

Hypervisor/OS can freely change VM Config Table

- No cache flushes
- No atomic updates
- No explicit movement of directory state
Second level: Protocol VH_A

**Directory** as Second-level Protocol

- Any tile can act as first-level directory
- Similar to DRAM directory
  - But: **blocking** to avoid deadlocks
  - How to track and name first-level directories?

Full bit-vector of sharers to name any tile

- State stored in DRAM
- Possibly cache on-chip

+ **Maximum flexibility**
- **DRAM State**
- **Complexity**
Second level: Protocol $VH_B$

**Broadcast** as Second-level Protocol

Attach **token count** for each block [token coherence]

- $T$ tokens for each block. One token to read, all to write
- Allows 1-bit at memory per block
- Eliminates system-wide ACK

+ Minimal DRAM State
  - e.g. saves 4GB of mem in a 32GB system !!

+ Enables easier optimizations
- Global coherence requires more activity
Result: Runtime

Eight VMs x Eight Cores Each = 64 Cores
(e.g. eight instances of Apache)
Questions and thoughts

- Is this scheme scalable to 1000-core CMP’s?
- Is content-based page sharing still a valid goal to pursue?
  - There might be a page shared by 10 VM’s
    - Duplicate private copies, vs.
    - Single shared copy
- Tile ↔ VM assignment: how to take communication patterns into consideration?
  - Contiguous tiles
    - Good for intra VM-sharing
  - Separated tiles
    - Good for inter VM-sharing