Achieving QoS in Server Virtualization

Intel Platform Shared Resource Monitoring/Control in Xen

Chao Peng (chao.p.peng@intel.com)
Increasing QoS demand in Server Virtualization

**Data center & Cloud infrastructure**
- Both public/private clouds call for it, especially for multiple tenants environment
- Key QoS requirements:
  - Service availability
  - Classes of service
  - Bandwidth / throughput

**Network Function Virtualization (NFV)**
- Intrinsic QoS demand from traditional ‘carrier-grade’ telecommunication environment
- Key QoS requirements:
  - Service availability
  - Classes of service
  - Bandwidth / throughput
  - Transmission delay / response time / interrupt latency
  - Low noise ratio and error detection
No QoS: Thread Contention

Full Contention (No QoS): CPU2006 29x29

Example on Haswell Client
SPEC Pairs +
Linux OS
Cache/Memory Contention

Data on Haswell Client (3GHz, 4 cores, 8MB cache, DDR3-1333, SPEC* CPU2006)

Resource contention causes up to 4X slowdown in performance
(Need ability to monitor and enforce cache/memory resource usage)
The fundamental reason for QoS in virtualization... is **Shared resource** contention.

- **Physical CPU**
  - Realtime scheduler
  - CPU pinning/isolation for high priority VM
- **Memory**
  - Cache aware memory allocation
  - Dedicated memory region for high priority VM
- **Shared device**
  - Dedicated device assignment
- **Platform shared resource**, such as LLC, memory bandwidth?
  - NO! Software can’t do anything for that.
Several Shared Resources (LLC, Memory Controller) cause execution contention.

- No performance guarantees when App’s / VM’s run in parallel
- Can be detrimental to Communications workloads (NFV / SDN)

What is Platform QoS?

1. Cache Monitoring and Allocation
2. Memory Bandwidth Monitoring
3. Future Features (Under Research)
Platform QoS Vision
Extending monitoring / enforcement across the platform

Cached

Code and Data Prioritization
Cache Allocation
Cache Monitoring

Memory Bandwidth Monitoring

Memory

Rich Set of Technologies Developed Along Cache and Memory Vectors
## Intel® Resource Monitoring/Control Technologies Roadmap

<table>
<thead>
<tr>
<th></th>
<th>2014</th>
<th>2015</th>
<th>2016</th>
</tr>
</thead>
</table>
| **Intel Xeon E5 v3 Family** | • Cache Monitoring Technology (CMT) | • Previous features, plus:  
  - Memory Bandwidth Monitoring (MBM)  
  - Cache Allocation Technology (CAT) | **Future Processors**  
  - Previous features, plus:  
  - Code and Data Prioritization (CDP)  
  - Future: Features under consideration, feedback solicited |
| **Intel Xeon D-1500 Family** | • Previous features, plus:  
  - Memory Bandwidth Monitoring (MBM)  
  - Cache Allocation Technology (CAT) | **Future Processors**  
  - Previous features, plus:  
  - Code and Data Prioritization (CDP)  
  - Future: Features under consideration, feedback solicited |

| **Xen 4.5** | • Cache Monitoring Technology (CMT) | **Xen 4.6**  
  - Cache Monitoring Technology (CMT)  
  - Memory Bandwidth Monitoring (MBM)  
  - Cache Allocation Technology (CAT) | **Xen 4.7**  
  - Cache Monitoring Technology (CMT)  
  - Memory Bandwidth Monitoring (MBM)  
  - Cache Allocation Technology (CAT)  
  - Code and Data Prioritization (CDP)  
  - More New Features available for Xen |

**Future Processors**
- Previous features, plus:
  - Code and Data Prioritization (CDP)
- Future: Features under consideration, feedback solicited

**Xen 4.7**
- Cache Monitoring Technology (CMT)
- Memory Bandwidth Monitoring (MBM)
- Cache Allocation Technology (CAT)
- Code and Data Prioritization (CDP)
- More New Features available for Xen
Platform Quality of Service:  
*Key concept -- RMIDs and CLOS*

- Resource Monitoring IDs (RMIDs) for the Monitoring features
- Classes of Service (CLOS) for the Allocation features
- Both organized as fields in IA32_PQR_ASSOC MSR

**Diagram:**
- Thread(s)
- Application(s)
- VM(s)
  - Flexible N:M Mapping
  - Classes of Service (CLOS) (Logical Construct)
  - Resource Monitoring IDs (RMIDs) (Logical Construct)

**IA32_PQR_ASSOC MSR (64b length):**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>63</td>
<td>Reserved</td>
</tr>
<tr>
<td>32</td>
<td>CLOS Field</td>
</tr>
<tr>
<td>31</td>
<td>Reserved</td>
</tr>
<tr>
<td>0</td>
<td>RMID Field</td>
</tr>
</tbody>
</table>

**Field Widths:**
- CLOS Field Width: \( \lceil \log_2(1 + \text{MaxCLOS}) \rceil \)
- RMID Field Width: \( \lceil \log_2(1 + \text{MaxRMID}) \rceil \)
High-Level Monitoring Usage Flow

**CMT Resource Association**
- On Context Switch*
  - RMID (for resource tracking)
  - PQR Register

**CMT Resource Marking**
- Application Memory Request
  - Tag with RMID
  - Memory Request

**CMT Resource Reporting**
- Per Application Monitoring Request
  - Event select for RMID
  - Read monitoring counter

---

**Cache Subsystem**
- QoS Aware Resource Monitoring
  - Set 1
  - Set 2
  - Set 3
  - ... ...
  - Set n
    - Way 1
    - ... ...
  - Way 16

**Example:**
- Occupancy, Mem BW

**PQR RMID identifies application for QoS resource-aware cache unit**
High-Level Allocation Usage Flow

**QoS Enum/Config**
- Enumerate CAT
- Configure class of Service w/ bitmasks

**QoS Association**
- On Context Switch
- Set Class of Service in PQR

**Allocation**
- Application Memory Request
- Tag with Cache Class of service

**Cache Subsystem**
- QoS Aware Cache Allocation
  - COS 1: Mask1
  - COS 2: Mask2
  - COS 3: Mask3
  - COS 4: Mask4

**Architectural Implementation dependent**

**Implementation dependent**

**Transaction**

**Enforce mask**

**Set 1**
- Way 1

**Set 2**
- Way 1

**Set 3**
- Way 1

**Set n**
- Way 1

**Way 16**
With CAT applied: Reduced Thread Contention

CAT Applied: 6MB Dedicated to HP App, LP Apps share 2MB

CAT Significantly reduces contention

Remaining contention due to memory bandwidth

Previous Contention Reduced Substantially!

Important Thread=6MB isolated, 3 Low-Priority threads share 2MB

<table>
<thead>
<tr>
<th></th>
<th>Max</th>
<th>Avg</th>
<th>Geomean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Contention</td>
<td>4.37x</td>
<td>1.37x</td>
<td>1.31x</td>
</tr>
<tr>
<td>CAT Applied</td>
<td>3.44x</td>
<td>1.25x</td>
<td>1.22x</td>
</tr>
</tbody>
</table>
Implementation in Xen - Overview

• Basic design for Xen
  ▪ Per-VM Monitoring/Allocation
  ▪ Mechanism in hypervisor, policy in user stack

• XL commands:

```bash
# xl help
...
psr-hwinfo Show hardware information for Platform Shared Resource
psr-cmt-attach Attach Cache Monitoring Technology service to a domain
psr-cmt-detach Detach Cache Monitoring Technology service from a domain
psr-cmt-show Show Cache Monitoring Technology information
psr-cat-cbm-set Set cache capacity bitmasks(CBM) for a domain
psr-cat-show Show Cache Allocation Technology information
```

More reference @ http://xenbits.xen.org/docs/unstable/misc/xl-psr.html
Implementation in Xen - Monitoring

Step 1: Check if CMT is supported

```bash
$ xl psr-hwinfo -cmt
Cache Monitoring Technology (CMT):
Enabled    : 1
Total RMID : 63
Supported monitor types:
cache-occupancy
total-mem-bandwidth
local-mem-bandwidth
```

Step 2: Attach RMID to domain

```bash
$ xl psr-cmt-attach <domid>
```

Step 3: Show monitoring data (LLC cache occupancy)

```bash
$ xl psr-cmt-show cache_occupancy
Total RMID: 63
Name                  ID  Socket 0  Socket 1
Total L3 Cache Size   20480 KB 20480 KB
Domain-0              0   14240 KB 14976 KB
ExampleHVMDomain      1   4200 KB  2352 KB
```

Step 4: Detach RMID from domain

```bash
$ xl psr-cmt-detach <domid>
```

MBM:

```bash
$ xl psr-cmt-show total_mem_bandwidth <domid>
$ xl psr-cmt-show local_mem_bandwidth <domid>
```
Implementation in Xen - Allocation

Step 1: Check if CAT is supported

```
$xl psr-hwininfo --cat
Cache Allocation Technology (CAT):
Socket ID : 0
L3 Cache  : 12288KB
Maximum COS : 15
CBM length : 12
Default CBM : 0xfff
```

Step 2: Set custom cache Bitmask for domain

```
$xl psr-cat-cbm-set <domid> 0xff
```

Step 3: Show domain LLC cache allocation

```
$xl psr-cat-show

ID     NAME        CBM
0       Domain-0    0xfff
1       HVM-guest  0xff
```

Step 4: Set default cache bitmask for domain

```
$xl psr-cat-cbm-set <domid> 0xfff
```

Next Slide: Besides xl commands(which display usage info on screen), let’s explore some benefits / use cases...
• **Metering and monitoring**
  - Any VM using more than X% of LLC: indicates heavy workload and node is busy.
  - All VM using less than X% of LLC: candidate for migration destination.

• **Noisy neighbor detection**
  - Define alarm for noisy neighbor, including cache threshold and alarm action.
  - Call alarm actions to do migration.
  - Use CMT filter to choose one node with enough available cache to do migration.
CAT improves NFV interrupt latency

*Significant improvements to interrupt-processing latency and predictability!*

Interrupt Latency with Aggressor Application (Fork Bomb) unpredictable with long tail

Cache Allocation Technology successfully contains the aggressor application

See more @ Cache Allocation Technology Improves Real-Time Performance
CMT/CAT feedback loop

- Utilize Monitoring (CMT or MBM) and Allocation (CAT) capabilities to build feedback loop to control efficient use of shared resources
- Optimize Cache and Memory Bandwidth allocation based on priorities
- A variety of throughput or prioritization-based algorithms have been shown to provide benefit depending on use model
Possible future work

• Feature Improvement
  - HW: More RMID/CLOS.
  - HW: Incremental feature improvements to reduce latencies.
  - SW: RMID recycling in Xen (Linux already has this).

• Future Features
  - Under consideration – *Driven by customer feedback!*

• Xen Scheduler improvement
  - [Intel Cache Monitoring: Current Status and Future Opportunities](#) by Dario Faggioli

• Guest emulation
  - Expose these technologies to guest so that guest and apps can take advantage of it.

• More usage models
  - In program or in end-user scenarios.
Additional Resources

- **Specification & Reference**
  - [Cache Monitoring and Cache Allocation Technology Landing page](#)
  - [Architecture Software Development Manual](#) (Vol 3b., Chapter 17.14 and 17.15, covers CMT, CAT, MBM, CDP)

- **Blogs**
  - [Part 1: Introduction to CMT](#)
  - [Part 2: Discussion of RMIDs and CMT Software Interfaces](#)
  - [Part 3: Use Models and Example Data](#)

- **NFV**
  - [NFV Delivers Packet Processing Performance with Intel](#)
  - [Cache Allocation Technology Improves Real-Time Performance](#)
  - [Packet Processing - Cache Monitoring Technology, Memory Bandwidth Monitoring & Cache Allocation Technology](#)

- **Xen**
  - [Intel Cache Allocation Technology(CAT) design for XEN](#)
  - [Intel Cache Monitoring: Current Status and Future Opportunities](#)
Thank you!