Resource Allocation: Intel Resource Director Technology (RDT)

Fenghua Yu <fenghua.yu@intel.com>

Aug 23, 2016
Introduction

• Intel Resource Director Technology (RDT)
  • Monitoring: Cache Monitoring Technology (CMT), Memory Bandwidth Monitoring (MBM), and more.
    • Passively monitor resources usage to identify QoS and performance bottlenecks
  • Allocation: Cache Allocation Technology (CAT), Code and Data Prioritization (CDP), and more.
    • Actively allocate resources to achieve better QoS and performance
Problems of Cache Sharing

Sometimes sharing is bad… Noisy Neighbor

Slow Down High Pri Process

High IRQ Latency in Real Time

Low Throughput in Containers
Solution?

No sharing.....Allocate separate cache for each app and no more noisy neighbor

Speed Up High Pri Process

Low IRQ Latency in Real Time

High Throughput in Containers
Cache Allocation Technology (CAT)

- Enables OS or Hypervisor or docker to specify the amount of cache space an app can use
- Enables dedicated cache space available for high priority apps.
- CAT L3 was first introduced on Haswell server, then on Broadwell server and Skylake server
  - L3 is LLC (Last Level Cache) on the processors
- CAT L2 is released in Software Development Manual and will be implemented on future x86 processors
Code and Data Prioritization (CDP)

• Extension of CAT

• Enables isolation and separate prioritization of code and data
  • provides separate code and data masks per CLOSID.

• First implementation is on Broadwell server and then on Skylake server
CAT L3 Hardware Architecture

L3_QoS_MASK_n MSRs per socket

<table>
<thead>
<tr>
<th>CLOSID</th>
<th>Cap Bit Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>CBM 0</td>
</tr>
<tr>
<td>1</td>
<td>CBM 1</td>
</tr>
<tr>
<td>2</td>
<td>CBM 2</td>
</tr>
</tbody>
</table>
Multi Resources Allocation: L2 and L3 CAT

<table>
<thead>
<tr>
<th>CLOSID</th>
<th>Cap Bit Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>CBM 0</td>
</tr>
<tr>
<td>1</td>
<td>CBM 1</td>
</tr>
<tr>
<td>2</td>
<td>CBM 2</td>
</tr>
</tbody>
</table>

L2_QoS_MASK_n MSRs per core

<table>
<thead>
<tr>
<th>CLOSID</th>
<th>Cap Bit Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>CBM 0</td>
</tr>
<tr>
<td>1</td>
<td>CBM 1</td>
</tr>
<tr>
<td>2</td>
<td>CBM 2</td>
</tr>
</tbody>
</table>

L3_QoS_MASK_n MSRs per socket
Enable Features in Linux Kernel

Switch to a process

Resource allocation schema for the process?

Y

Get closid in schema per cpu

Update closid in PQR msr only for new closid

Run the process with closid in PQR and cbm in QoS_MASK

N

Get closid in schema per task_struct

Feature enumeration and enabling from CPUID

User interface defines resource allocation schemata per task or per CPU
User Interface

• Kernel creates a file system “resctrl” (standing for ReSourCe ConTRoL) to hold user interface

• Mounted as /sys/fs/resctrl
  • Upon mounted, the file system has the directory:
    • /sys/fs/resctrl/info/…: contains info for resource allocation
    • /sys/fs/resctrl/tasks: contains all pids initially
    • /sys/fs/resctrl/cpus: all 1’s initially
    • /sys/fs/resctrl/schemata: all 1’s initially means all processes and CPUs can use all resources by default

• User can create resource allocation schema for a group of tasks or cpus
  • mkdir /sys/fs/resctrl/part_1: Create a L3 partition in sub-dir part_1 under the resctrl file system.
    • /sys/fs/resctrl/part_1/tasks: user can add pids to the file to assign resources on the pids. Initial value is empty.
    • /sys/fs/resctrl/part_1/cpus: user can add cpu masks to the file to assign resources on the cpus. Initial value is 0.
    • /sys/fs/resctrl/part_1/schemata: user can write L3 and L2 cbms to this file. Initial value is all 1’s.

• User modifies schemata, and assigns tasks/cpus to use the schemata.
Open Stack specifies L3 allocation schema and starts Guest1 and Guest2

Kernel resctrl interface

Kernel assigns guest a portion of L3

L3 Allocated for Guest1 Allocated for Guest2
Usage case 2 - CAT L3 Isolation for Containers

- Docker
  - libcontainer
  - Kernel resctrl interface
  - Kernel assigns one portion of L3 for a container

- L3
  - Allocated for Container1
  - Allocated for Container2

- Hardware
- User space
- Kernel space
Usage case 3 - CAT L3 for Real Time Apps

Sysadmin launches real time app p1

Sysadmin specifies L3 allocation schemata for p1 and p2

Kernel resctrl interface

Kernel assigns app a portion of L3

Allocated for p1

L3

Allocated for p2

Sysadmin launches real time app p2
Resource Allocation Tool

- User space resource allocation tool
- Based on resctrl kernel interface
- Graphic User Interface
- High level info hiding HW details
- Will open source once done
Performance Improvement by CAT (Case 1)

**Intel® Resource Director Technology (Intel® RDT) - University of California, Berkeley**

- Network functions are executing simultaneously on isolated core's, throughput of each Virtual Machines is measured
- Min packet size (64 bytes), 100K flows, uniformly distributed
- LLC contention causes up to 51% performance degradation in throughput

**Table: Max.% throughput degradation, normalized**

<table>
<thead>
<tr>
<th>Software</th>
<th>Max.% Throughput Degradation</th>
</tr>
</thead>
<tbody>
<tr>
<td>EndRE 2</td>
<td>4.2%</td>
</tr>
<tr>
<td>IPsec</td>
<td>12.6%</td>
</tr>
<tr>
<td>Suricata</td>
<td>22.2%</td>
</tr>
<tr>
<td>Snort</td>
<td>33.3%</td>
</tr>
<tr>
<td>Stats</td>
<td>46.1%</td>
</tr>
<tr>
<td>MazuNAT</td>
<td>51%</td>
</tr>
<tr>
<td>LPM</td>
<td>27.6%</td>
</tr>
<tr>
<td>Firewall 2</td>
<td>12.8%</td>
</tr>
<tr>
<td>Efficuts</td>
<td>6.2%</td>
</tr>
</tbody>
</table>

Without Cache Allocation Technology

Source: University of California, Berkeley

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. Configurations: see slide 31. For more complete information, visit [http://www.intel.com/performance/datacenter](http://www.intel.com/performance/datacenter).
Performance Improvement by CAT (Case 1)(cont.)

Intel® Resource Director Technology (Intel® RDT) - University of California, Berkeley

- Network functions are executing simultaneously on isolated core’s, throughput of each Virtual Machines is measured
- Min packet size (64 bytes), 100K flows, uniformly distributed
- VM under test is isolated utilizing CAT, 2 Ways of LLC are associated with the Network function. Isolation only causes ~2% variation

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. Configurations: see slide 28. For more complete information, visit http://www.intel.com/performance/datacenter
Performance Improvement by CAT (Case 2)

No QoS: Thread Contention

Full Contention (No QoS): CPU2006 29x29

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

Low-Priority Applications
(Identical Instances):
- aster
- bzip2
- cactusADM
- calculi
- dealii
- gemm
- Gromacs
- hpcg
- hpl
- hmmer
- libquantum
- Leslie3d
- mc
- mgrid
- nomira
- perlbench
- perfume
- sjeng
- soplex
- soplex
- tetgen
- vips
- vips
- xalancbmk

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

High Priority Application

App Slowdown vs. Alone

Resource contention causes up to 4X slowdown in performance
(Need ability to monitor and enforce cache/memory resource usage)

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

App Slowdown vs. Alone

Previous Contention Reduced Substantially!
Status

• Previous cgroup user interface Linux kernel patches were rejected by upstream because of cgroup and user interface design limitations.

• We proposed a new extendable and fine tuning user interface infrastructure design
  • Multi resources allocation: extendable to L3, L2, and so on.
  • Per resource domain allocation: fine control each resource allocation unit
  • Resource allocation for kernel thread:
    • Allocation per CPU:

• The new resctrl user interface and kernel design patches were published on lkml on 7/12/2016 and are being reviewed by the community.

• We will release v2 patch set once it’s done base on the community’s comments.
References

• x86 Software Developer Manual
• Latest patches for CAT and CDP: [https://github.com/fyu1/linux/tree/cat16.1](https://github.com/fyu1/linux/tree/cat16.1) and lkml
Acknowledgements

Tony Luck, H. Peter Anvin, Ravi Shankar, Vikas Shivappa, Sai Praneeth Prakhya from Intel Open Source Technology Center
Q & A