Network Based Coherency: Extending a Processor's Coherency Domain over a Standard Network

Bob Quinn, Isam Akkawi, Krishnan Subramani, Shahe Krakirian
August 2008
Why Network Based Coherency?

Current data center problems
- Low server utilization
- Rigid resource configurations with slow and expensive moves, adds & deletes

Solution – Scalable servers with dynamically provisioned CPU, memory & IO resources

Datacenter Resource Pool

Guest OS
Cache Coherency Evolution from a Bus to a MicroNetwork

Integrated Hardware Support for Virtual Machine Monitors

- Nested page tables
- Fast switching between VMM and guest
- Virtual interrupt support
- VMM intercept of selected events or instructions

![Diagram showing cache coherency evolution and integrated hardware support for virtual machine monitors.](image)
Next Step: Coherent Network

Network used as coherent interconnect fabric for servers
Hardware provides Coherent Distributed Shared Memory view & low-level I/O

Software virtualizes CPU, Memory, and IO to the OS/App
The Coherent Network Controller

Physical Servers (nodes)

Coherent Network Controller

X86

DRAM

cHT and QPI

Infiniband and Ethernet

Switch

Fabric
**Coherent Network Controller: 3Leaf TL1550**

- **Dual coherent HT ports**
  - Up to 8GB/sec total bandwidth per port (16b * 1GHz DDR each direction)

- **Hardware managed cache coherency**

- **Manages up to 1TB of memory**

- **Dual IB/Ethernet fabric ports**
  - 4x InfiniBand DDR & SDR (up to 20 Gbps each direction), Version 1.2 compliant.
  - 802.3 10GBASE-CX4 (up to 10Gbps each direction, 20Gbps also supported)
  - Integrated Serdes: No external PHY required

- **Reliable delivery across fabric**
Coherent Transport: Reliable Delivery Protocol (RDP)

Requirements for running coherent memory transactions over fabric
- Short packets → Need very low transport overhead
- Guaranteed in order end-to-end delivery
- Multiple paths for high availability
- Layer-2 agnostic
- Capability to share fabric with non-coherent transactions (e.g. IO)

3Leaf Reliable Delivery Protocol
- Very light weight transport
- Runs over IB link layer or Ethernet MAC layer – low loss fabric
- Automatic retry on fabric errors -- error recovery in micro-seconds
- Automatic path failover
- Supports Multiple transaction level protocols
  - Coherent Memory protocol
  - Messaging
  - DMA
  - Reliable multicast
- Multiple Virtual channels
RDP Packet Overhead

**RDP Header**

<table>
<thead>
<tr>
<th>DWORD</th>
<th>Bit</th>
<th>Transmit first</th>
<th>Transmit 2(^{nd})</th>
<th>Transmit 3(^{rd})</th>
<th>Transmit last</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0x3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 0x3 Ver = 0
- AP VC
- CmdCode
- SendRN
- SendSN_LSB
- DstLNID
- SrcLNID

**RDP over InfiniBand Packet**

<table>
<thead>
<tr>
<th>DWORD</th>
<th>Bit</th>
<th>Transmit first</th>
<th>Transmit 2(^{nd})</th>
<th>Transmit 3(^{rd})</th>
<th>Transmit last</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(n+1)</td>
<td>(n+2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(n+3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Transmit first
- Transmit 2\(^{nd}\)
- Transmit 3\(^{rd}\)
- Transmit last

- IB Local Route Header (LRH)
- RDP Header
- RDP Payload (0 to 47 DWORDs)
- RDP Acks (1 to 2 DWORDS)
- IB Invariant CRC
- IB Variant CRC

**RDP Ack**

<table>
<thead>
<tr>
<th>Bit</th>
<th>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>AckSN</td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

- --
- AckRN
- Ver

**RDP over Ethernet Packet**

<table>
<thead>
<tr>
<th>DWORD</th>
<th>Bit</th>
<th>Transmit first</th>
<th>Transmit 2(^{nd})</th>
<th>Transmit 3(^{rd})</th>
<th>Transmit last</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(n+1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(n+2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(n+3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(n+4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Transmit first
- Transmit 2\(^{nd}\)
- Transmit 3\(^{rd}\)
- Transmit last

- Preamble
- Ethernet Header
- RDP Header
- RDP Payload (0 to 47 DWORDs)
- RDP Acks (1 to 2 DWORDS)
- Ethernet FCS
- Ethernet FCS

**IB + RDP overhead: 28 bytes**
- Includes SOP & EOP

**Ethernet + RDP overhead: 54+ bytes**
- Frame overhead (Headers, Acks, FCS): 34 bytes, pad to 64 bytes if needed
- Preamble + IFG: 20 bytes
- Manages cache coherent Distributed Shared Memory across multiple nodes

- TL1550 behaves like another x86 CPU socket on the node
  - For local request to remote memory -> like CPU Memory
  - For remote nodes to local (Home) memory -> like another CPU

- 144MB Node Cache
  - Reduces remote memory latency

- Hardware managed line caching
- Software allocated, hardware managed page-caching
Latency Management

- OS level ccNUMA awareness
- Page placement (3Leaf Software)
- Page replication (3Leaf Software)
- Page caching (3Leaf Software + TL1550)
- Line caching (TL1550)
- Coherence acceleration (TL1550)
144MB Node Cache optimizes latency

- **Node Cache caches remote data**
  - Data stored externally (DDR SDRAM)
  - Tags stored internally (SRAM)

- **Line cache**
  - 64B cache-line granularity
  - 8 way set associative
  - Hardware managed MOSI coherency

- **Page cache allows software to manage caching**
  - 4KB page granularity
  - 4 way set associative
  - Hardware manages coherency
  - Modified lines tracked in line cache Software allocates/replaces
  - Hardware assist for software heuristics
  - Page Search Engine to identify hot pages
  - Per-page R/W bit for page replacement
  - Performance monitoring counters to monitor hit rates
TL1550 enforces coherency across fabric

1) uP issues read to remote memory
   • Misses in local cache

2) uP sends request to TL1550

3) TL1550 misses in node cache and sends request to remote home node

4) Remote home node sends request to home uP

5) No other nodes using requested cache line so cache line is read from uP memory

6) Remote data returned
What is Fast Invalidate (FI)?

- **Fast stores to shared data**
  - Commit stores to shared data before it is globally visible
  - Preserves processor read-write ordering
  - Low latency store to shared data

- **Per-page configuration**

- **Lockdown** – Fast store from local CPU

- **Blocking** – Fast invalidate from remote CPU

- **Delayed Blocking**
  - Delay between FI response and blocking
  - Utilizes fabric delay to overlap HT invalidate latency
Fast Invalidate – Lockdown

- FI response to local Req before remote sharers have been invalidated
- Blocks local data until the remote sharers have been invalidated
- Data from remote nodes to local node is not affected by this Lockdown
TL1550

- FI response to before the CPU caches are invalidated
- Blocks remote data until the local CPU caches have been invalidated
- Data from the local node to the remote nodes is not blocked

X86

CPU Probes

CPU Rsp

FI Data Barrier

Probe

Data

FI Rsp

Fabric
Fast Invalidate Lockdown with Enhanced Blocking
Fast Invalidate – Pros and Cons

**Lockdown – Fast store from local CPU**
- **Pros**
  - Latency of store to shared data is similar to store to local memory
  - CPU does not “see” latency of remote invalidates
  - TL1550 preserves x86 read-write ordering semantics
- **Cons**
  - Subsequent remote requests could be delayed due to Lockdown

**Blocking – Fast invalidate from remote CPU**
- **Pros**
  - Overlaps invalidates from remote nodes with switch latency
  - Impact of Blocking in Remote node reduced by Enhanced Blocking
  - TL1550 preserves x86 read-write ordering semantics
- **Cons**
  - Subsequent remote requests could be delayed due to Lockdown
110 mm² in TSMC 90GT

- **Dual Coherent HT Interfaces**
- **Reliable Delivery Manager**
- **Coherent Memory Manager**
- **SDRAM Controller**
- **DMA & Int Manager**
- **Dual 20 Gbs serdes**
  - Proven technology
  - Multi-rate, 2x4 Channels
  - IB (SDR, DDR)
  - Ethernet (XAUI, DDR XAUI)
TL1550 Technology Summary

- **Technology**: TSMC 90 nm GT
- **Operating Frequency**: 400 Mhz
  - Standard ASIC design flow
- **Gate count**: 6 M gates
- **SRAM bits**: 24.8 Mb (Repairable)
- **Total IO**: 607
- **Die Size**: 10.7 mm x 10.3 mm
- **Package**: 1207 OLGA
- **Compatible with Torrenza socket (Socket F)**
  - Standard for Opteron and Opteron-based co-processors
- **Power**: <20W