MEMSCALE™: a new way of architecting memory in clusters

F. Silla
Technical University of Valencia
Spain

Third Symposium of the HyperTransport™ Center of Excellence, Mannheim, February 8-9th, 2011
Outline

• Introduction
• Keeping the trend ...
• MEMSCALE: breaking the trend ...
• A 64-node prototype based on FPGAs
• Performance analysis
  • Applications
  • In memory databases
• MEMSCALE: exclusive vs. shared memory
  • Shared memory for in-memory databases
  • Shared memory for parallel applications
• MEMSCALE: power saving opportunities
• Future improvements
• Conclusions
Introduction

How would the “perfect” large computing installation look like?

1 socket, up to ~24 GB RAM
2 sockets, up to 64-128 GB RAM
4 sockets, up to 512 GB RAM
8 sockets, up to 80 cores, up to 1 TB RAM
Introduction

How would the “perfect” large computing installation look like?

Following this trend, the perfect large computing installation would aggregate as many computing cores as needed and as much memory as required under a single memory address space.
Following this trend, the perfect large computing installation would aggregate as many computing cores as needed and as much memory as required under a single memory address space.

... but large computing deployments do not look like that.
How actually do large computing installations look like?
Introduction

• Clusters, either current or past, are just the aggregation of **many independent computers** interconnected by some network technology.
Introduction

Clusters

- As many OSes as computers
- No coherency among computers
- Memory at nodes is overscaled (but idle) in order to satisfy any application
- Message-passing programming model
  - Harder to program than regular shared memory
    - Explicit communication
  - Message-passing systems
Introduction

Shared-memory computers

- Powerful single machines:
  - Up to 80 cores
  - Up to 1TB of RAM
- Parallel programming:
  - Shared-memory programming model
  - Easy to program
- Cache coherency among processors

Message-passing systems

- Really large clusters
- As many OSes as computers
- Memory at nodes is overscaled (but idle) in order to satisfy any application
- Message-passing programming model
  - Harder to program
    - Explicit communication
  - No coherency among computers
Introduction

Shared-memory computers

- Powerful single machines:
  - Up to 80 cores
- Easy to program
- Cache coherency among processors

Scalability concerns !!!!

Message-passing systems

- Really large clusters
- Memory at nodes is overscaled (but idle) in order to satisfy any application
- Message-passing systems
  - Explicit communication
  - No coherency among computers
  - As many OSes as computers

Scalability is kept!!!!

... something in between?
Keeping the trend ...

... in between: large coherent shared-memory systems

- Based on aggregating several commodity computers
  - Provide a single coherent domain
  - One operating system
- Software aggregation:
  - vSMP by ScaleMP
  - Symmetric Computing
  - vNUMA
- Hardware aggregation:
  - NumaConnect by NUMASCALE
  - Bullx Supernode
  - HP Superdome 2
  - SGI Altix UV
Keeping the trend ... vSMP by ScaleMP

- As it is a software solution, presumably based on the page fault mechanism
  - The same mechanism leveraged for carrying on regular swap to disk
  - Memory pages are exchanged among nodes
  - More complex than remote swap, as coherency must be kept
- Scales up to 128 nodes
  - 1024 cores, 64TB RAM memory
- Based on Infiniband
  - Reliability
  - Performance
- Price:
  - Depends on sockets/cores per motherboard and amount of shared memory:
    - $3,500 for quadcore 4-socket motherboards
    - $231,500 for 64 nodes (16 cores per node) and 2TB of RAM
Keeping the trend … SGI Altix UV

• Scales up to 128 dual-socket motherboards
  • 2048 cores, 16TB RAM memory
• Based on the UV HUB ASIC chip and QPI
• Fat-tree topology
• 1.2 microseconds remote memory access time (1-hop away)
  • Additionally, 100 nanoseconds per hop
Keeping the trend ... NumaConnect

- Scales up to 4096 nodes
  - 65536 cores and 256TB RAM memory
- Based on the NumaChip ASIC chip and HyperTransport
- 1D, 2D, or 3D torus topology
- 1 microsecond remote memory access time (1-hop away)
  - In addition, 1.5 microsecond for the coherency layer
  - Additionally, 40 nanoseconds per hop
Keeping the trend ...

<table>
<thead>
<tr>
<th>BASED ON SW</th>
<th>BASED ON HW</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SCALEMP</td>
<td>SGI Altix UV</td>
<td>CHIP IN MAINBOARD</td>
</tr>
<tr>
<td>Symmetric Computing</td>
<td>Bullx</td>
<td></td>
</tr>
<tr>
<td>vNUMA</td>
<td>NumaScale</td>
<td>CHIP IN ADD-ON CARD</td>
</tr>
</tbody>
</table>
Aggregating x86-based commodity computers into a single coherent domain provides a huge amount of resources.

But a penalty for sharing resources is paid:

- System size is limited by the coherency protocol
- Performance is limited by the coherency protocol

**Is really needed the overall coherency protocol?**

Most shared-memory parallel applications do not scale beyond a few tens of cores.

However, many applications (parallel or sequential) may still benefit of large amounts of memory:

- Some scientific applications and simulations
- In-memory databases
MEMSCALE: breaking the trend …

- Sharing memory across the cluster but not sharing cores neither caches
- The overhead of the coherency protocol among nodes is avoided
MEMSCALE: breaking the trend ...

<table>
<thead>
<tr>
<th>BASED ON SW</th>
<th>BASED ON HW</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCALEMP</td>
<td>SGI Altix UV</td>
</tr>
<tr>
<td>Symmetric Computing</td>
<td>Bullx</td>
</tr>
<tr>
<td>vNUMA</td>
<td>NumaScale</td>
</tr>
<tr>
<td></td>
<td>MEMSCALE™</td>
</tr>
</tbody>
</table>
MEMSCALE: breaking the trend ...

- Currently based on HyperTransport, but also possible over QPI or PCIe
- Accessing remote memory is completely done by hardware
MEMSCALE: working principle

Node #0 (Source)
Here: Issuing loads/stores on remote memory

Node #1 (Target)
Here: serving as memory host

EXTOLL: Custom FPGA-based high performance interconnection network, although IB or ETH also possible
MEMSCALE

- Memory map of a node in a 255-node cluster
  - Each node has 16 GB … but may access up to 4 TB

- Reservation procedure relies on software
A 64-node prototype based on FPGAs
A 64-node prototype based on FPGAs

- Virtex-4 FX100 RMC Prototype
  - 156MHz core clock
  - HT400 interface (1.6GB/s)
  - 6 links, each 624 MB/s
- Each node:
  - 4x AMD Opteron 2 GHz Quad Core
  - 16GB RAM
  - Standard Linux
Remote memory access latency

- 20 million consecutive accesses to an 8-byte integer array located in remote memory
- Each hop adds 600 ns (two way)

Current design is a memory mapped I/O unit one outstanding request
Latency prediction for improved technology

The use of wider data paths has not been considered in the plots

Performance analysis
PARSEC performance

Performance analysis
SPLASH performance

Performance analysis
SPLASH performance

Performance analysis
In-memory databases

Performance analysis
In-memory databases

Performance analysis
In-memory databases

Performance analysis
MEMSCALE: exclusive vs. shared memory
MEMSCALE: shared memory for in-memory DB

in-memory database

Node A
Node B
Node C
Node D
MEMSCALE: shared memory for in-memory DB

MEMSCALE (16 nodes only)

MySQL Cluster

MEMSCALE: exclusive vs. shared memory
MEMSCALE: shared memory for parallel computing

The underlying hardware does not provide coherency among nodes:

- Coherency, when required, must be provided by the software
  - Enhanced barriers and locks → applications remain unchanged
MEMSCALE: shared memory for parallel computing

Barrier performance

Work currently in progress

MEMSCALE: exclusive vs. shared memory
MEMSCALE: power saving opportunities

Distributed parallel applications

- Fast nodes would provide computing resources
- Medium or slow nodes would provide memory resources
MEMSCALE: power saving opportunities

In-memory databases

16 slow-core node

MEMSCALE: power saving opportunities
MEMSCALE: power saving opportunities

In-memory databases

Regular memory

MEMSCALE: power saving opportunities
Future improvements

HTX-Board v1.3
- XC4VFX100
- HT400 (1.6GB/s)
- Internal data path
  - 32bit @ 156MHz (624MB/s)
- 624MB/s links

HTX-Board v2.0
- XC6VLX240T
- HT600 (2.4GB/s)
- Internal data path
  - 64bit @ 300MHz (2.4GB/s)
- 2.4GB/s links
Conclusions

• Shared-memory systems are preferred over message passing ones
  • Shared memory is easier to program
  • Shared memory avoids having to partition the data
• Large shared-memory deployments are not feasible
  • The coherency protocol does not scale, introducing unaffordable overhead
• MEMSCALE provides shared memory while avoiding the coherency overhead. **Aims at keeping the best of shared memory without its scalability concerns**
• Performance numbers obtained with a slow FGPA technology
  • Feasibility of the idea proven
  • Tremendous benefits for some applications (large in-memory databases)
  • Large power saving opportunities
More people in the MEMSCALE team:

- José Duato
- Hector Montaner
- Knut Kujat
- Javier Prades
- Ricardo Marín
- Holger Fröning
Questions?

Thanks!