

Dynamic Partitioned Global Address Spaces for Improving Memory Efficiency

Jeff Young

Advisor: Sudhakar Yalamanchili

Computer Architecture and Systems Lab
School of Electrical and Computer Engineering
Georgia Institute of Technology

December 3rd, 2008



Outline

- ① Problem Statement
- ② Trends
 - Networking and DRAM Memory
 - Pin Bandwidth
 - Network Interface/ Memory Controller Integration
 - Research Overview
- ③ DPGAS
 - PGAS Overview
 - DPGAS
 - Hypertransport over Ethernet Bridge
 - Research Questions
- ④ Current Results
 - Experimental Setup
 - Scale Up Evaluation
 - Scale Out Evaluation
- ⑤ Future Research

Challenges

- Large datacenters are cost and energy inefficient
 - Statistics suggest that 1.5% of all U.S energy costs go to servers and datacenters and this cost could double by 2011¹
 - Memory is a large contributor of overall cost and energy²
- A lack of memory sharing between server blades leads to overprovisioning of memory per blade
- Enterprise workloads are often time-varying, which lead to dynamic needs for memory allocation
 - Simultaneous peak memory demand is less than the sum of applications' peak memory footprints

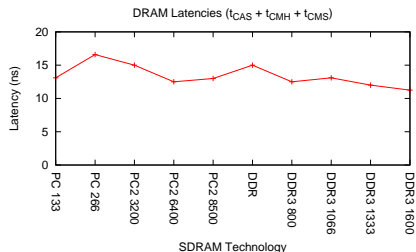
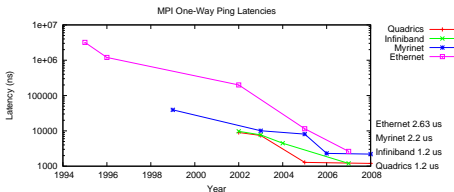
¹EPA Report to Congress on Server and Data Center Energy Efficiency, 2007

²C. Lefurgy, et al, Energy Management for Commercial Servers, IEEE Computer 2003

Important Trends Affecting Memory Systems

- Network vs. Memory
- Pin Bandwidth
- Integration of Interconnect and Memory Controller

Networking and Memory Trends



- Link and switch cut-through latencies dropping faster than DRAM latencies
- We already see commodity board level, multi-socket coherent, NUMA machines
- Lesson: Memory reach can be extended beyond board with small performance penalties

Memory Pin Bandwidth

- Trends
 - Core scaling
 - Slower growth in pin bandwidth ³
 - Consolidation via virtualization \Rightarrow increased memory pin bandwidth demand
- Memory bandwidth/core decreasing
- Lessons:
 - Use network pins to increase (remote) memory BW
 - Need more pins \Rightarrow 3D stacks, network-on-package, serial memory controllers (FB-DIMM)

Network Interface/ Memory Controller Integration

- Network interfaces now integrated on chip — HT, QPI, PCIe
 - Hardware distance from the wire to the memory controller greatly reduced
 - Memory functions can be designed to take advantage of low-latency network access
- However, point to point links are not switched - two options for inter-blade communication:
 - Custom interconnects to extend reach of point-to-point links
 - Commodity interconnects (Ethernet, Infiniband) used with encapsulation
 - Network latency and cost trends indicate commodity interconnects are a good choice
- Lessons:
 - Integrate memory controllers and network interfaces
 - Integrate memory function and communication

Memory Systems Research Philosophy

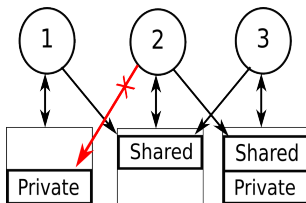
- Focus on making DRAM-to-DRAM transfers transparent and fast
- Use memory sharing to create a global non-coherent managed *physical* address space
 - Ensure address space portability across generations
 - Emulate a 64-bit address space for current processors
 - Rethink system view as a network of memory controllers
- Layer customized behaviors on these mechanisms
 - Region based memory semantics (e.g. for coherence)
 - Explicit sharing
 - Push vs. pull messaging

Focus of this talk:

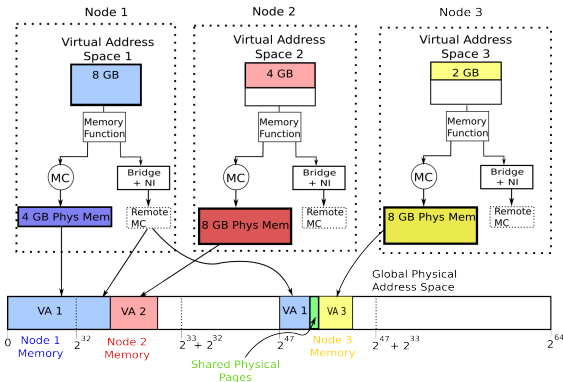
- Architecting a system-wide non-coherent physical address space

Partitioned Global Address Space

- Concept of private local memory and shared global memory ^a
- Current approaches are software based and SPMD (subset of MIMD)
- **Languages:**
 - UPC - LLNL, Berkeley, GWU
 - Titanium - Berkeley
 - DARPA HPCS languages
 - X10 - IBM
 - Chapel - Cray
 - Fortress - Sun
- **Messaging layers:**
 - **Gasnet** - Berkeley low-level put/get model based on Active Messages



Dynamic Partitioned Global Address Space

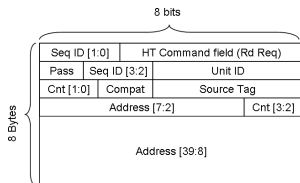


- Flexible assignment of physical address space to memory controllers
- Protection issues handled by virtual memory system
 - Bridge mapping handled and coordinated by OS

Hypertransport Overview

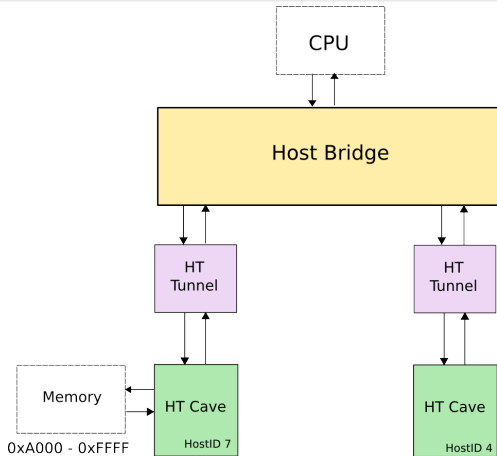
- Hypertransport, QPI and PCI Express are all candidates to use with DPGAS
 - HT and QPI are both point-to-point, packetized and support virtual channels
- HT is open-source and currently available \Rightarrow was used for initial implementation
- What are some of the challenges for using localized, point-to-point interconnects over Ethernet?
 - First, how does HT work normally?

HT Terminology



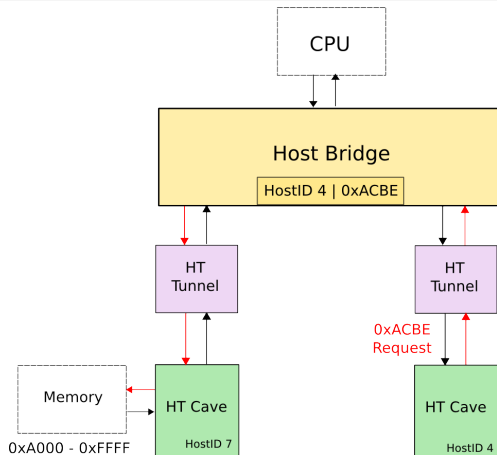
- **Control and Data packets** - request, response, broadcast are control packets; the data attached are data packets
- **Posted** - does not require a response; writes can be posted
- **Nonposted** - requires a response in the form of data or Target Done; reads or writes can be nonposted
- **Response** - control and data packets that are sent in reaction to a request
- **SeqID** - used to identify packets that are strongly ordered (receive in order)
- **SrcTag** - used to identify all transactions from a host - up to 32 at a time
- **UnitID** - identifies the source of a transaction in local HT
- **HT Cave** - end point device that can send/receive HT requests/responses
- **HT Tunnel** - passive device that passes HT packets between bridges and end points

Hypertransport Cave Example



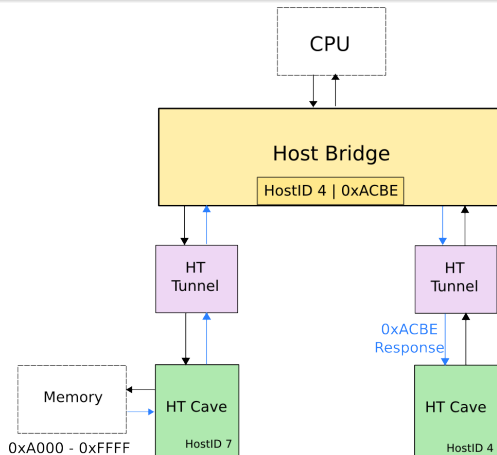
How is a local Hypertransport packet sent/received?

Hypertransport Cave Example



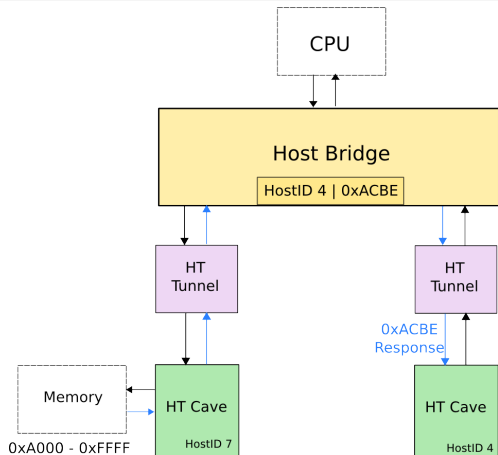
Host bridge buffers information about request.

Hypertransport Cave Example



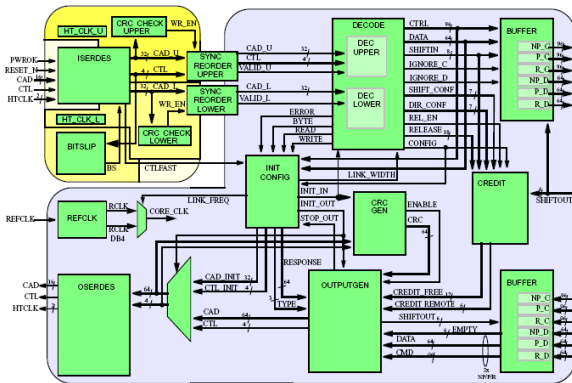
Host bridge matches response with request to send data downstream.

Hypertransport Cave Example



How do we extend this to multiple nodes?

Heidelberg HT Core



Heidelberg cave device is used for local HT signals; bridge component handles remote encapsulation. ^a

^a<http://ra.ziti.uni-heidelberg.de/coeht/>

Hypertransport over Ethernet Demonstration Bridge - Statistics

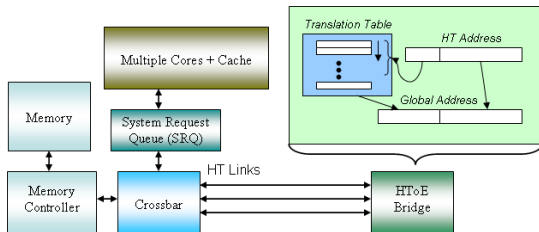


Figure: HToE Bridge in relation to Opteron Memory Subsystem ⁴

- Low latency - bridge encapsulation takes 24 - 40 ns
- Bridge \Rightarrow 1300-1500 FPGA slices; HT cave \Rightarrow 5222 slices on Virtex 4 FX60⁵

⁴P. Conway, B. Hughes, *The AMD Opteron Northbridge Architecture*, IEEE Micro, 2007

⁵D. Slognat, et al, *An open-source HyperTransport core*, ACM Transactions on Reconfigurable Technology

Hypertransport over Ethernet Demonstration Bridge - Approach

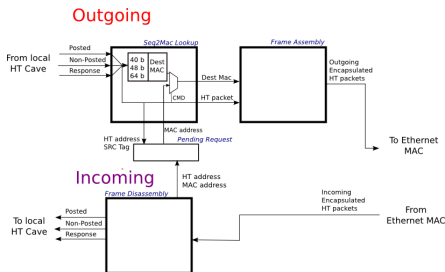


Figure: Bridge Organizational Model

- Modular approach and encapsulation allows software to be portable as processor physical address space grows.
 - Extension from the 40-bit to 64-bit physical address
 - Creation of a HyperTransport packet which includes a 64-bit extended address
 - Map the most significant 24 bits of destination address to a 48-bit MAC address and encapsulation into an Ethernet frame.

Some DPGAS Research Questions

- Memory allocation
 - How can virtualized hosts allocate memory effectively?
 - Hypervisor level profiling of application page swapping, balloon drivers reallocate memory on need basis
 - Abstractions for physical memory allocation
 - Memory leases: time-based or explicitly managed
 - Cooperative spilling, e.g. with caches
- Memory consistency models for integrated NI/MC
 - Relaxed protocols might be most suitable
 - Dependence on interconnect
- How can DPGAS be used to reduce memory power and cost?
 - Allows for efficient sharing between blades and reduces peak provisioning requirements
 - This is the focus of our recent investigations.

Evaluation of DPGAS Impact on Cost/Power Efficiency

- **Question:**

- How much can effective memory sharing strategies impact DRAM power and cost?

- **Approach:**

- Trace based simulation used to generate profiles of page fault behavior vs. memory footprint
- Analytical models used to evaluate power and cost of DRAM memory and reductions for HP server configurations
 - Performance statistics (page faults) obtained from page table simulation
 - Cost statistics obtained from HP's server business website ⁶
 - Power statistics obtained from HP power calculators ⁷

⁶ HP Proliant DL servers - cost specifications, <http://h18004.www1.hp.com/products/servers/platforms/>,

2008

⁷ HP Power Calculator Utility,

Experimental Setup - Trace generation

- Sampled from high frequency and large footprint program regions
 - Gprof: Execution time distribution
 - Heap profilers and Linux ps command were used to find location of maximum memory footprint
- 2.1 billion references were taken from these points using Simics infrastructure ⁸
 - Simics traces included background operating system traffic
 - 100 million references were used to warm the page table simulation
- Page table simulation evaluated 7 physical memory sizes (32 MB - 2 GB) for each application

⁸ P. Magnusson, *Simics: A Full System Simulation Platform*, Computer, 2002

Experimental Setup - Server Workloads

Model	CPU Cores	Max. Physical Memory	Base Server Cost/Power
HP Proliant DL785 G5	8 quad-core 2.4 GHz Opterons	512 GB	~\$42,000/1110 W
HP Proliant DL165 G5	2 quad-core 2.1 Ghz Opterons	64 GB	~\$2,000/197 W

- Server configurations were selected to be high-end (8 CPU server) and low-end (2 CPU server) - workloads fit into either 64 GB (high-end) or 16 GB (low-end of memory)
- Combinations of each benchmark were used to model independent workloads running in VMs⁹

VM instances - Proliant 785 G5								
Blade Number	1	2	3	4	5	6	7	8
DIS	20	18	16	14	12	20	16	12
SSCA	18	16	14	12	20	12	10	15
MCF	16	14	12	20	18	11	8	18
MILC	14	12	20	18	16	9	14	9
LBM	12	20	18	16	12	13	12	21
Total	80	80	80	80	80	65	60	75

⁹S. Chalal, et al, Memory Sizing for Server Virtualization, Intel TR, 2007

Experimental Setup - Memory Allocation

- Preliminary Results:
 - Static allocation of memory for each workload for normal and DPGAS configurations
 - Allocation based on

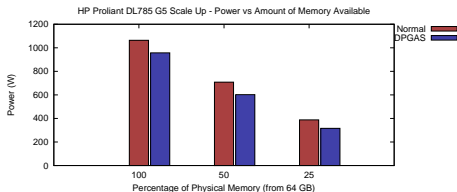
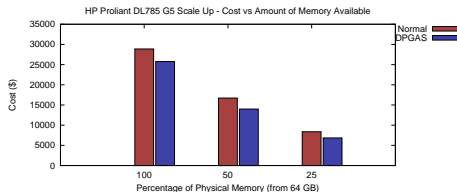
$$\frac{\% \text{ reduction in page faults}}{\text{amount of memory allocated}}$$

- Other memory allocations also could be used:
 - Memory balancing (libnuma), Berkeley's Firehose
 - Borrow from cooperative caching - spill/receive model
 - Memory leases - timed allocation based on need and cost metric

Experimental Setup - Analytical Models

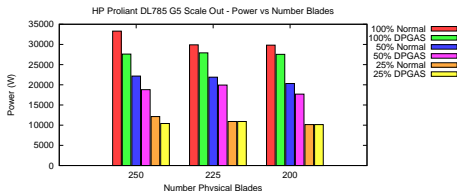
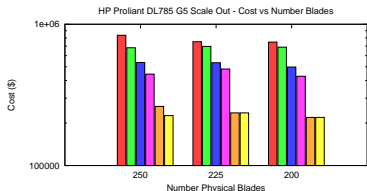
- *Scale up* (overprovisioning server memory) and *Scale out* (adding additional servers to accommodate large workloads) approaches were evaluated
- Scale up
 - 8 servers were evaluated using memory throttling to evaluate effects of overprovisioning
 - Ex: An overprovisioned server might require 64 GB of memory. How well does this server perform with only 32 or 16 GB of memory?
- Scale Out
 - 200, 225, and 250 servers were evaluated to show effects of consolidation and memory throttling
 - 250 high-end servers could feasibly support 19,500 VMs; low-end servers could support 4,700 VMs
 - What are the cost/power/performance tradeoffs for consolidating this workload onto 200 servers?

HP Proliant 785 G5 Scale Up - Power and Cost Savings



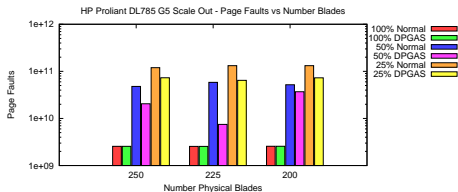
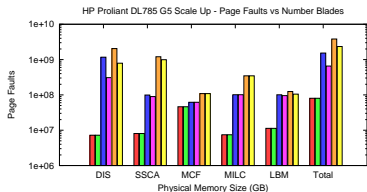
- DPGAS memory sharing allowed for 10% to 19% power savings and 11% to 18% cost savings
- For 8 servers, this translates to saving \$1,536-\$3,108 and 72 to 107 Watts just by using memory sharing
- DPGAS allows for sharing beyond blade boundaries, reducing memory fragmentation

HP Proliant 785 G5 Scale Out - Power and Cost Savings



- Blade consolidation and memory throttling analyzed for eight core servers
- Cost savings were 1% to 18% and power savings were from 7% to 17%
- For 250 blades, this translates into \$154,000 savings in memory cost and 5,650 Watt savings in input power.

HP Proliant 785 G5 - Performance



- Focusing on cost and power is a tradeoff with performance
- 50 % memory throttling increases page faults by 19x with normal allocation but only 8x with DPGAS
- Consolidation onto 200 blades with 50 % base memory increases page faults 20x and 14x, respectively
- DPGAS can not only save cost and power, can also improve performance of throttled/consolidated systems

More Results

- Low-end server showed similar results for performance and cost/power savings
- Commodity DIMMs reduce overall cost and power savings, as does amount of DRAM supported
- Performance-constrained analysis showed that by keeping cost and power usage constant, performance of throttled systems could be improved by DPGAS
 - 50% memory throttling resulted in 6x as many page faults with normal allocation, 3x as many with DPGAS
- For more information see MS Thesis, “DPGAS for High-Efficiency Computing”

Related Work

- Memory Efficiency
 - Memory Miser - Tolentino, Cameron
 - Adaptive memory throttling, scheduling
- PGAS
 - Software approaches - UPC, X10, Titanium
 - Gasnet
 - RDMA - Liang '05 is the best example of low-level implementation
- Power and Cost Analysis
 - HP labs, 2008 ISCA paper
 - Google
 - Feng - Green Destiny

Future Research Questions

- Integration of Memory Systems and Networks
- Matching on-chip and off-chip network semantics
 - Message ordering constraints
 - Interaction with flow control can induce deadlock
 - Flow control and error recovery
 - Switched links and local on-chip segments networks may use different techniques
 - On-chip QPI/HT/PCIe must be beefed up?
 - Traditional bridging issues and optimizations
- Multiple off-chip interfaces?
- Building NICLess systems!

Questions?

- Any Questions?
- More info at: <http://www.ece.gatech.edu/research/labs/casl/hec.html>

Percent Memory Allocation Pseudocode

```

While free memory is left
  For each server
    Reset winner and % threshold
    For each VM workload running on that blade
      If % improvement for VM  $\geq$  % improvement of other VMs
        Set VM as winner and set % improvement as new threshold
    If winner exists
      Allocate memory via DPGAS to the winning VM
    Else if no winners
      Allocate memory locally as needed
  
```

Return