Horus: Large-Scale Symmetric Multiprocessing for Opteron Systems

Horus lets server vendors design up to 32-way Opteron systems. By implementing a local directory structure to filter unnecessary probes and by offering 64 Mbytes of remote data cache, the chip significantly reduces overall system traffic as well as the latency for a coherent HyperTransport transaction.

Rajesh Kota
Rich Oehler
Newisys Inc.

Apart from offering x86 servers a migration path to 64-bit technology, the Opteron processor from AMD enables glueless eight-way symmetric multiprocessing (SMP). The performance scaling of important commercial applications is challenging above four-way SMP, however, because of the less-than-full interconnection. Interconnect wiring and packaging is severely taxed with an eight-way SMP system.

Scaling above an eight-way SMP system requires fixing both these problems. The Horus application-specific IC, to be released in third quarter 2005, offers a solution by expanding Opteron's SMP capability from eight-way to 32-way, or from 8 to 32 sockets, or nodes.[1] As the "Work on Symmetric Multiprocessing Systems" sidebar shows, many SMP implementations exist, but Horus is the only chip that targets the Opteron in an SMP implementation.

In a *quad*—a four-node Opteron—Horus acts as a proxy for all remote CPUs, memory controllers, and host bridges to local Opteron processors. The chip extends local quad transactions to remote quads and enables requests to remote quads. Key to Horus's performance is the chip's ability to cache remote data in its remote data cache (RDC) and the addition of Directory, a cache-coherent directory that eliminates the unnecessary snooping of remote Opteron caches.

For enterprise systems, Horus incorporates features such as partitioning; reliability, availability, and serviceability; and communication with the Newisys service processor as part of monitoring the system's health.

In performance simulation tests of Horus for online transaction processing (OLTP), transaction latency improved considerably. The average memory access latency of a transaction in a four-quad system (16 nodes) with Horus running an OLTP application was less than three times the average memory access latency in an Opteron-only system with four Opterons. Moreover, as the number of CPUs per node increased, improvements became even more significant.

Horus architecture

Each Opteron[2,3] comprises an integrated on-die memory controller; a host bridge that

Published by the IEEE Computer Society

provides the interface between the processor's coherent domain and noncoherent domains (I/O); and three HyperTransport (HT) links.[4] The three HT links provide glueless SMP to eight nodes, each of which can have up to four units: one for the memory controller, one for the host bridge, and two for the two CPU cores. If there is only one CPU core, one of the units remains unused. The memory controller is dual ported and supports double-data-rate synchronous DRAM (DDR-SDRAM).

In an SMP system that uses multiprocessor-enabled Opterons, physical memory extends across memory controllers, with a particular controller becoming home to a range of physical addresses. Each Opteron could have an I/O chain connected to its host bridge. Each processor has address-mapping and routing tables. The address-mapping table maps nodes to physical memory or the I/O region. The routing table maps HT links to nodes for routing HT packets.

Horus's cache coherence (CC) protocol, which is atop the coherent HT (cHT) protocol, lets designers merge multiple Opteron quads into a larger, low-latency, cache-coherent system. The CC protocol provides functions beyond the cHT protocol, including remote data caching, a cache-coherent directory, optimized quad-to-quad protocols, and a quad-to-quad delivery mechanism that guarantees packet delivery by retrying when soft errors occur.

Figure 1 shows the inter-

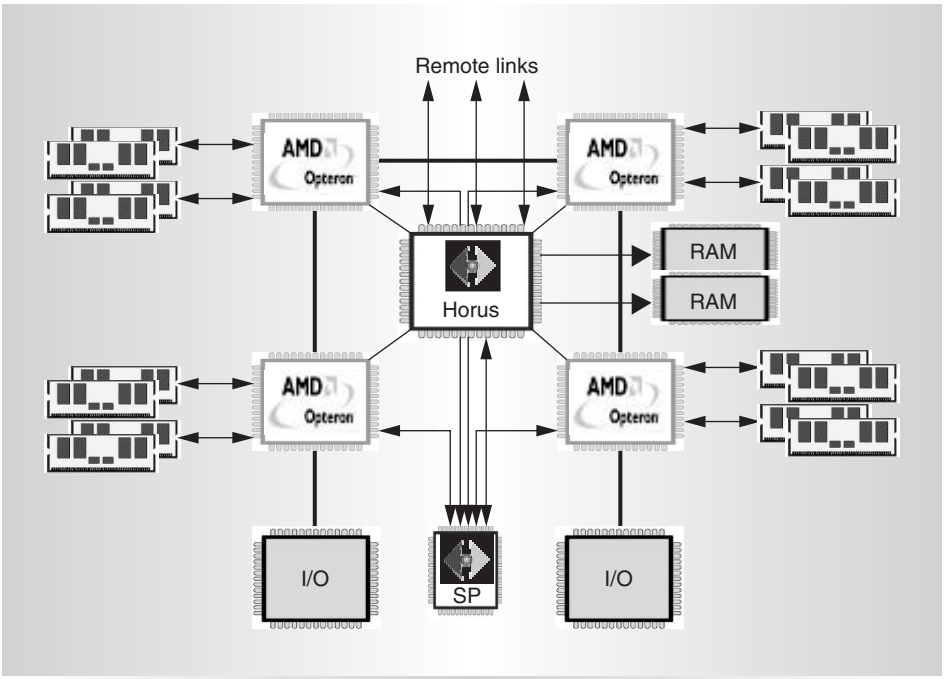

Figure 1. Interconnecting four Opteron processors and Horus to form a quad. The service processor (SP) also interacts with Horus.

Work on Symmetric Multiprocessing Systems

Both research and industry have produced SMP systems. Some have used proprietary high-end processors; others have used commercial x86 processors. Horus is the only SMP system to use Opteron processors. The following list hints at the diversity of implementations:

- A. Agarwal et al., "The MIT Alewife Machine: Architecture and Performance," *Proc. 22nd Int'l Symp. Computer Architecture*, IEEE CS Press, 1995, pp. 2-13.
- T. Brewer and G. Astfalk, "The Evolution of the HP/Convex Exemplar," Proc. Computer Conf., IEEE CS Press, 1997, pp. 81-86.
- A. Charlesworth, "Starfire: Extending the SMP Envelope," *IEEE Micro*, vol. 18, no. 1, Jan.-Feb. 1998, pp. 39-49.
- E. Hagersten and M. Koster, "WildFire–A Scalable Path for SMPs," *Proc. Int'l Symp. High-Performance Computer Architecture*, IEEE CS Press, 1999, pp. 172-181.
- J. Kuskin et al., "The Stanford FLASH multiprocessor," *Proc. Int'l Symp. Computer Architecture*, IEEE CS Press, 1994, pp. 302-313.
- "HP Integrity Superdome Server Technical White Paper," Hewlett Packard, Dec. 2004. http://h71028.www7.hp.com/ERC/downloads/5982-9836EN.pdf
- J. Laudon and D. Lenoski, "The SGI Origin: A ccNUMA Highly Scalable Server," Proc. 24th Ann. Int'l Symp. Computer Architecture, ACM Press, 1997, pp. 241-251.
- D. Lenoski et al., "The Stanford DASH Multiprocessor," *Computer*, vol. 25, no. 3, Mar. 1992, pp. 63-79.
- T. Lovett and R. Clapp. "STiNG: A CC-NUMA Computer System for the Commercial Marketplace," *Proc. Int'l Symp. Computer Architecture*, IEEE CS Press, 1996, pp. 308–317.
- S. Mukherjee et al., "The Alpha 21364 Network Architecture," *Proc. 9th Symp. High-Performance Interconnects*, IEEE CS Press, 2001, pp. 113-118; also in *IEEE Micro*, vol. 22, no. 1, Jan.-Feb. 2002, pp. 26-35.
- A.K. Nanda et al., "High-Throughput Coherence Control and Hardware Messaging in Everest," *IBM J. Research and Development*, vol. 45, no. 2, pp. 229-244, Mar. 2001.
- M. Woodacre et al., "The SGI Altix 3000 Global Shared-Memory Architecture," SGI, 30 Apr. 2003.

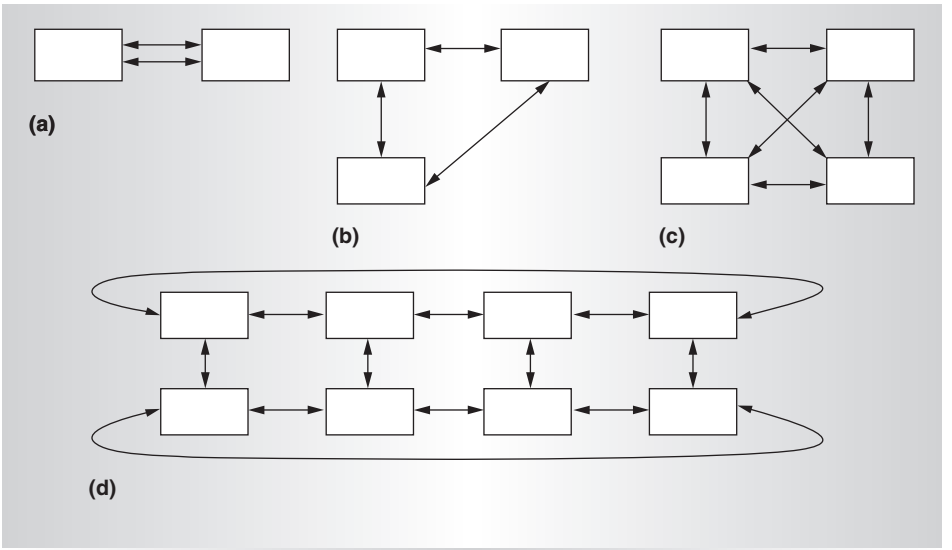

Figure 2. Configurations using Horus, with each square representing a four-node Opteron, or quad: two- (a), three- (b), four- (c), and eight- (d) quad systems.

connections between Horus and four Opteron processors within a quad.

The Horus chip has four cHT links as well as three remote links that interconnect the quads in different clock and power domains using Infiniband cables of up to 10 feet.[5] Each cHT link comprises 16 bits of data at 2 GHz, along with source-synchronous clocks that support a total data rate of 32 Gbps. The chip implements each remote link using 12 lanes of 3.125-GHz serializers and deserializers in a physical protocol layer with 8-bit/10-bit encoding, effectively yielding a data rate of 30 Gbps. The CC protocol can handle up to eight quads in various configurations, as Figure 2 shows.

Address mapping

To the local Opterons in a quad, Horus looks like another Opteron. Horus acts as a proxy for all the remote memory controllers, CPU cores, and host bridges. Newisys BIOS programs the mapping tables in the local Opterons to direct to Horus any requests to physical memory or I/O residing in the remote quads. The local Opteron is not aware of the remote Opterons.

As we described earlier, in a quad that uses Horus, each active memory controller has a contiguous range of physical addresses, as does each quad. The local Opteron's address-mapping tables assign to Horus the physical-address region above and below the quad's address region. The global address-mapping tables in Horus contain information about which quad is assigned what physical-address regions.

Packet retagging

Each cHT transaction consists of multiple packets from the various Opterons. The packet type could be requests, probes, broadcasts, or responses.[2-4] Some packets have associated data; some do not. All packets that belong to one transaction have a unique and common transaction ID so that the Opterons can stitch together all the packets related to a particular transaction.

Transactions that an Opteron in one quad generates could have the same transaction ID as a completely different transaction that another Opteron in a different quad generates. When a transaction goes through Horus from the local to the remote domain (or from the remote to the local domain), Horus creates a new transaction ID and substitutes it for the incoming transaction ID. Each Horus maintains a one-to-one mapping between transaction IDs in the local domain to transaction IDs in the remote domain.

Remote probing

Figure 3 shows a request from the local processor to local memory (LPLM) with remote probing, in which Horus extends the cHT protocol for a request from the local CPU to the local memory controller (MC). A local Opteron processor (L) includes the CPU and its caches, the MC, and the host bridge. The probe (P) is Opteron's cache memory snoop. All nodes that receive a probe (Horus and the Opterons) send a response to the source CPU or target MC, depending on the request type.[2,3] The response is either a probe response (PR) or read response (RR) if the line in the Opteron cache was dirty.

Remote fetching

Figure 4 shows a transaction from a local processor to remote memory (LPRM) using

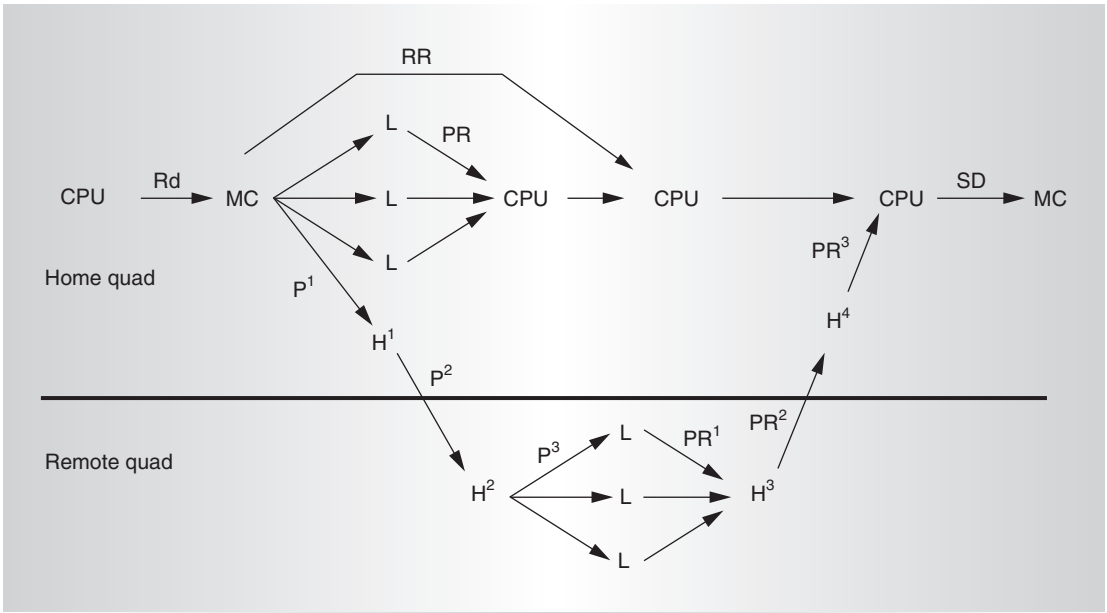

Figure 3. How a local-processor-to-local-memory (LPLM) request with remote probing works. Horus at the home quad (H^1) receives a probe (P^1) for a local-memory request and forwards that probe (P^2) to all remote quads. In this example, there is only one remote quad. The superscript for the probe represents different transaction IDs in different domains. The solid black line separates the home and remote quads. The Horus at the remote quad (H^2) receives remote probe P^2 and broadcasts probe P^3 to all local nodes. The remote-quad Horus accumulates CPU probe responses PR^1, and H^3 forwards the accumulated response (PR^2) back to the home quad Horus (H^4), which accumulates responses PR^2 from the remote quads and forwards the accumulated response (PR^3) back to the requesting CPU. The H with superscripts indicates a Horus at various points along the transaction. Read request (Rd), and read response (RR), which contains data, are other transactions; Probe responses contain only status information, not data. Source done (SD) indicates that the transaction has been committed at source.

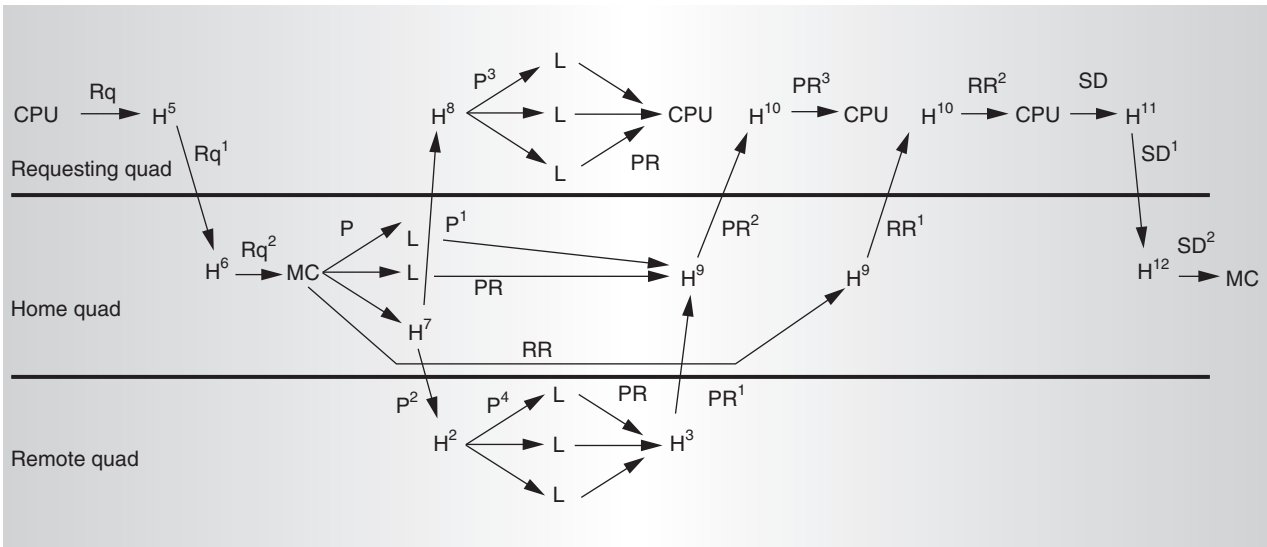

Figure 4. How a transaction from a local processor to remote memory (LPRM) with remote fetching works. The superscripts on the H, which designates the Horus, represent the Horus at various points in the transaction. Each quad, which the solid lines delineate, has only one Horus.

remote fetching. Horus at a home quad (H^5) accepts a local processor (CPU) request (Rq) that targets remote memory and forwards request Rq^1 to the home quad's Horus (H^6). H^6 receives Rq^1 and forwards request Rq^2 to the home quad's MC. The MC generates and forwards the probe (P) to home nodes, including the home quad Horus (H^7). The home quad Horus in turn forwards P to the remote quad (P^2) and the requesting quad (P^1).

Horus at the requesting quad (H^8) accepts probe P^1 and forwards probe P^3 to local nodes. The remote-quad Horus (H^2) accepts probe P^2 and forwards probe P^4 to local nodes. The remote-quad Horus (H^3) then accumulates responses and forwards the accumulated response (PR^1) to the home quad. Horus at the home quad (H^9) accumulates responses from local nodes and the remote quad (PR^1). Once Horus at the home quad has received all PRs, it forwards the accumulated response (PR^2) to the requesting quad (H^{10}). Horus at the home quad (H^9) also forwards the MC response to the requesting quad (RR^1).

At the requesting quad, the Horus (H^{10}) accepts PR^2 and forwards response PR^3 to the requesting quad (CPU). The requesting-quad Horus (H^{10}) also accepts response RR^1 from the MC and forwards RR^2 to the requesting quad (CPU).

After the requesting CPU receives all the responses, it completes the transaction by generating an SD response to Horus (H^{11}). The requesting-quad Horus (H^{11}) forwards response SD^1 to the home-quad Horus (H^{12}), which in turn forwards SD^2 to the MC, and the transaction is complete.

Horus implements these protocols through protocol engines (PEs), of which there are three types. The *local-memory PE (LMPE)* handles all transactions directed to local memory controllers and host bridges. The *remote-memory PE (RMPE)* handles all transactions directed to remote memory controllers and remote host bridges. The *special-function PE (SPE)*, which has access to the internal-control-register bus, processes peripheral component interconnect (PCI) configurations.

Performance enhancements

Although the features just described are essential to extending Opteron's SMP capabilities, it is equally important to address bandwidth and latency issues in a large SMP system.

Directory

Horus implements *Directory,* a CC directory, inside LMPE. Directory maintains invalid, shared, owned, and modified states for each local memory line that remote Opterons cache. It also maintains an occupancy vector, 1 bit per quad, to track which quad has a cached copy of the memory line. This tracking enables Horus to disallow probes to quads where a memory line wouldn't be cached, which helps reduce probe bandwidth and transaction latency.

Horus uses repairable SRAM to implement an on-die directory with a sparse two memory lines per entry (two sectors) and an eight-way, set-associative tag array. Sectoring limits the physical size of on-chip tag arrays. Sparcity—the ratio of the directory's total memory lines to the size of the caches in remote quads (including those in remote Horuses)—is 50 percent for a four-quad system. This percentage was the result of trading off die size and optimal performance in a four-quad implementation.

Directory allocates entries as Horus receives requests from remote nodes. It deallocates entries if the line is no longer remotely cached or if Horus must evict the least recently used entry to accommodate a new request. Directory issues a zero-sized write request to an evicted memory line to force invalidation or the flushing of dirty data from remote caches back to memory.

Figure 5 shows a request of a memory line from a local processor to a local memory controller that is not remotely cached. In this case, Horus looks up the memory line in the directory and finds that no remote quad is caching it. It therefore does not broadcast any remote probes, and the transaction completes quickly in the local quad. The figure shows the performance advantage that Directory affords.

Remote data cache

Horus supports 64 Mbytes of off-chip remote data cache (RDC), a limit that aims to keep Directory sparcity at 50 percent for four-quad systems. The RMPEs have on-chip tags to track data cached in off-chip memory. The tag array, which Horus implements using on-chip SRAM, has two memory lines per entry

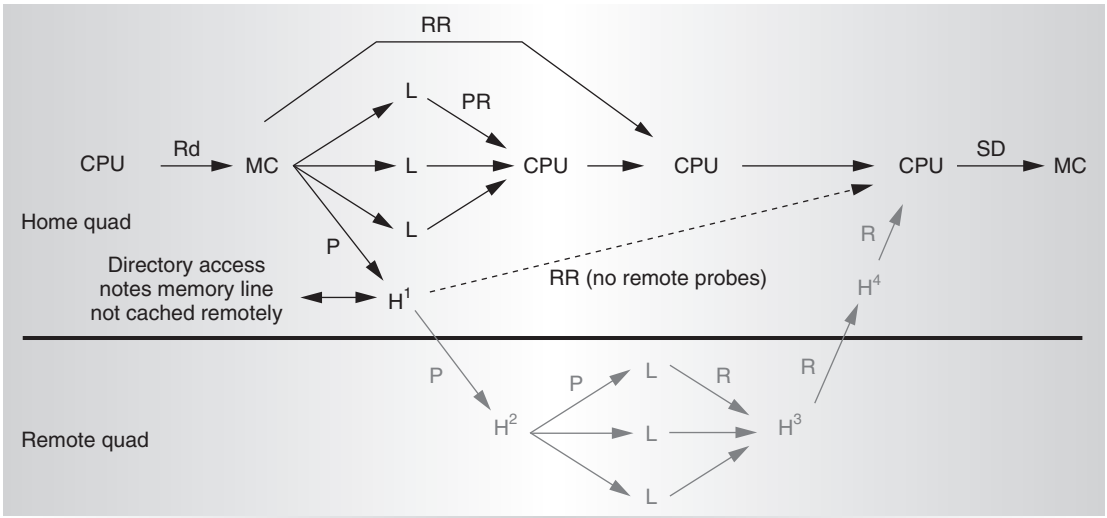

Figure 5. How Horus's Directory handles a memory line request. In this case, the request is from a local processor to a local memory controller that is not remotely cached, so the transaction completes locally. In gray are the packets that Directory has eliminated during this transaction, which gives some idea of how Directory reduces transaction latency.

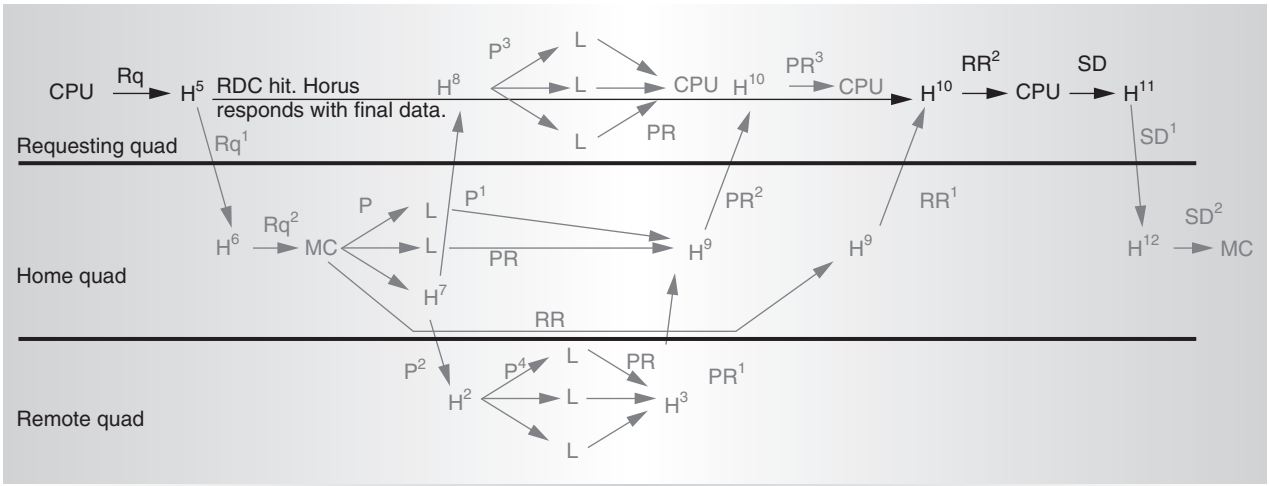

Figure 6. A hit in RDC for remote data request causes the transaction to complete locally. The RDC hit eliminates all the packets in gray, which shows the level of performance improvement.

(two sectors), is eight-way set-associative, and has single-cycle access. The RMPE maintains shared, exclusive, and invalid states for memory lines that the RDC holds.

Horus fills RDC in two ways. First, for remote requests that miss in RDC, the remote protocol keeps a copy of the data that the memory controller returns. Second, when a local Opteron evicts a dirty memory line from its cache, the RMPE keeps a copy of it. During conflict, the RDC implements the least recently used policy to evict memory lines.

Opterons forward requests for remote physical data to the RMPE. For shared requests, RMPE checks the tag array to see if the data is in the RDC. If it is, the RMPE forwards the data to the requesting Opteron and the transaction completes locally. As Figure 6 shows, RDC, like Directory, enables enormous performance gains by eliminating packets. It forwards requests for write permission of the memory line to the remote memory controller so that it can generate invalidated probes to all caches that have a copy of the memory line.

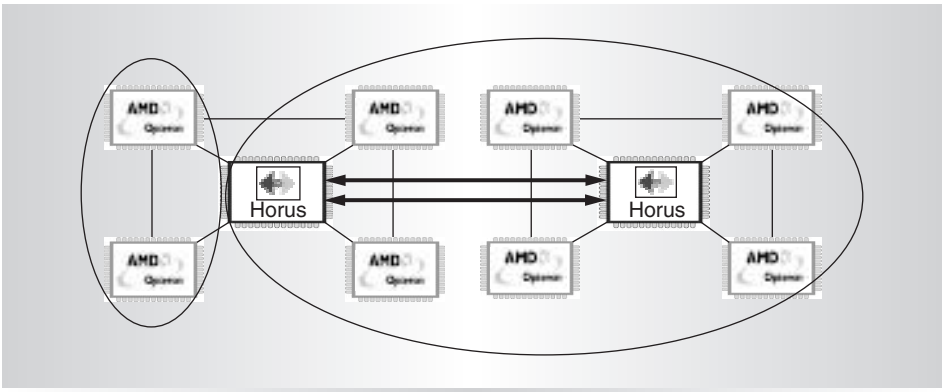

Figure 7. System with two unequal partitions. Dynamic system partitioning is possible along any remote links, but Horus can reside in only one partition at a time.

Partitioning

Horus, together with Newisys system management, lets designers partition a large-scale SMP system into smaller SMP systems. Partitioning has only two requirements: Each partition must have at least one I/O chain with a south bridge, and Horus can reside in only one partition at a time; transactions from multiple partitions cannot share a Horus.

Together, Horus and Newisys system management permit dynamic system partitioning along any remote links, but not across a cHT link, which requires resetting the Opterons. Only static partitions are allowed across cHT links. Figure 7 shows a two-quad, eight-node system in two unequal partitions.

Microarchitecture

The cHT transactions consist of command and data packets, which take different paths through Horus, as Figures 8 and 9 show.

cHT receiver and transmitter links

The cHT receivers and transmitters in Horus comply with the cHT protocols. Receiver modules decode incoming command and data packets. Data packets remain inside the data buffer until transmitter links or the RDC request them. Receiver modules forward command packets to the PE for processing. The cHT receiver and transmitter modules maintain complete separation between packets of different virtual channels specified in cHT.

Remote receiver and transmitter links

As we described earlier, Horus implements three remote links, and only three, to limit I/O pin count and still provide full connectivity for up to four quads.

We have extended the cHT protocol for remote links and separated the remote protocol layer from the remote link layer. The remote link layer is extremely reliable and implements a guaranteed-exactly-once delivery system. Inside Horus is the hardware support to enable hot plug and hot unplug of the remote links. The software must guarantee that no active transactions are in the system during hot unplug.

Pipelined protocol engines

Horus processes cHT packets through PEs, which let Horus process transactions simultaneously and thus provide more bandwidth. Each PE is pipelined and has a 32-entry-deep pending buffer, with one entry per transaction. Each transaction can comprise multiple cHT packets, so the pending buffer can keep track of the transaction state and the accumulating responses. Each PE can handle 32 transactions simultaneously.

Horus implements the cHT protocols in microcode, with the base protocol implemented in a ROM structure. A RAM structure also exists in the PE so that the BIOS can load a completely new protocol. As we described earlier, the BIOS programs a PE to be either a LMPE, RMPE, or SPE—which is actually the same physical PE instantiated multiple times. Each Horus has two RMPEs, two LMPEs, and one SPE.

Crossbars

As Figures 8 and 9 show, Horus has three crossbars, all of which are nonblocking. The *receiver* crossbar moves command packets from seven receiver links to five PEs. The *transmitter* crossbar moves command packets from five PEs to seven transmitter links. Data packets don't go through a PE. When they come into Horus, it buffers them in the receiver link. When a PE processes the command packet associated with a data packet, it either discards the data or forwards it through a transmitter link.

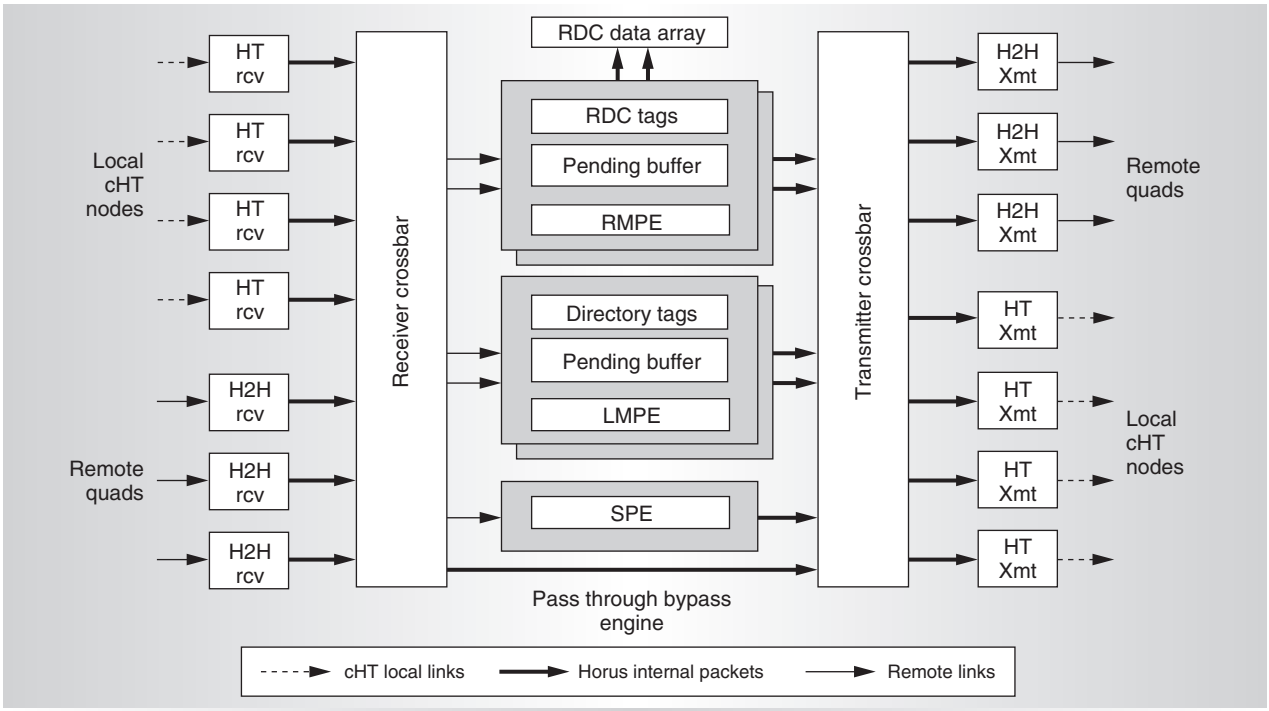

Figure 8. Path that command packets take inside Horus. The figure labels Horus-to-Horus transfers as H2H. HT is the HyperTransport protocol, and cHT is a coherent HT protocol. RDC is the remote data cache, RMPE is the remote memory protocol engine, LMPE is the local memory protocol engine, and SPE is the special function protocol engine.

After processing a command packet, the PE sends it to a transmitter link. The transmitter link requests data associated with the command packet from the *data* crossbar. The data crossbar moves data from the receiver links directly to the transmitter crossbar.

Bypass engine

The bypass engine forwards packets directly from receiver links to transmitter links without any modification. This is a very low-latency cut-through path. The bypass engine is so called because it bypasses packets both in local (between Opterons) and remote (between Horuses) domains. Bypass paths are completely nonblocking. When Opterons in a local quad don't have a direct cHT link between them, the local bypass path in Horus serves as that link. Because we limited Horus to three remote links, in configurations with more than four quads, remote bypass links route packets between quads that aren't directly connected.

Reliability, availability, and serviceability

Horus is ideally suited for an enterprise SMP system. On the remote links, Horus

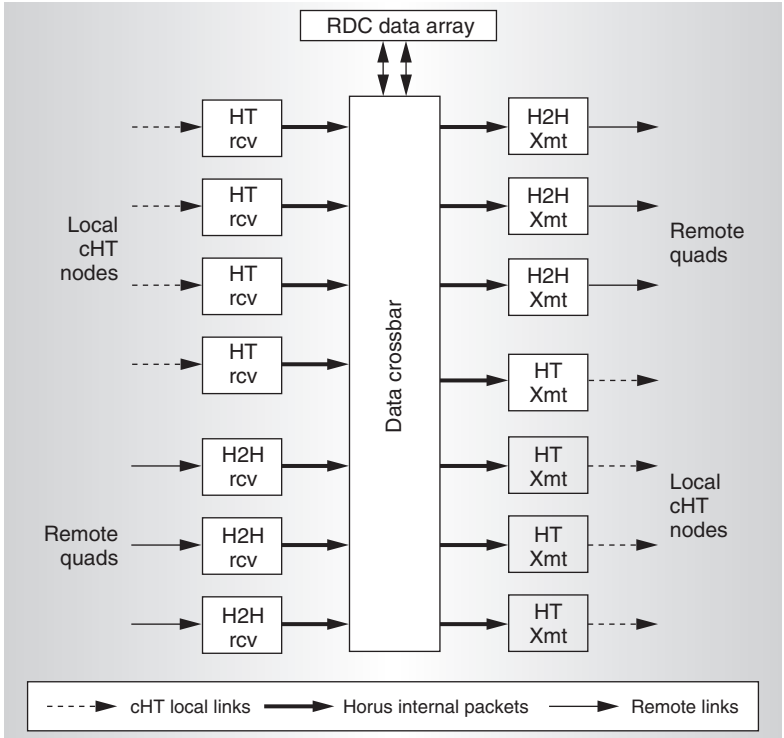

Figure 9. Path that data packets take inside Horus.

Table 1. Latencies in a Horus-based system with different configurations for different CPU-to-memory-controller requests.

Transaction type	Latencies (in ns) in two-, four- and eight-quad systems 2	3	4	8	Outcome and subsequent action
L2SM*	269	269	269	334	Directory hit: Probe all remote quads.
L2LM	293	293	293	357	Directory hit: Probe all remote quads.
L2RM	356	396	396	461	Remote data cache (RDC) miss and Directory hit in home quad: Probe all remote quads.
L2SM	96	96	96	96	Directory miss: No probes to remote quad.
L2LM	122	122	122	122	Directory miss: No probes to remote quad.
L2RM	139	139	139	139	RDC hit: Transaction completes locally.

*L2SM represents CPU requests to a memory controller in the same Opteron; L2LM, CPU requests to memory controller in a different Opteron; and L2RM, CPU requests to memory controller in a remote quad.

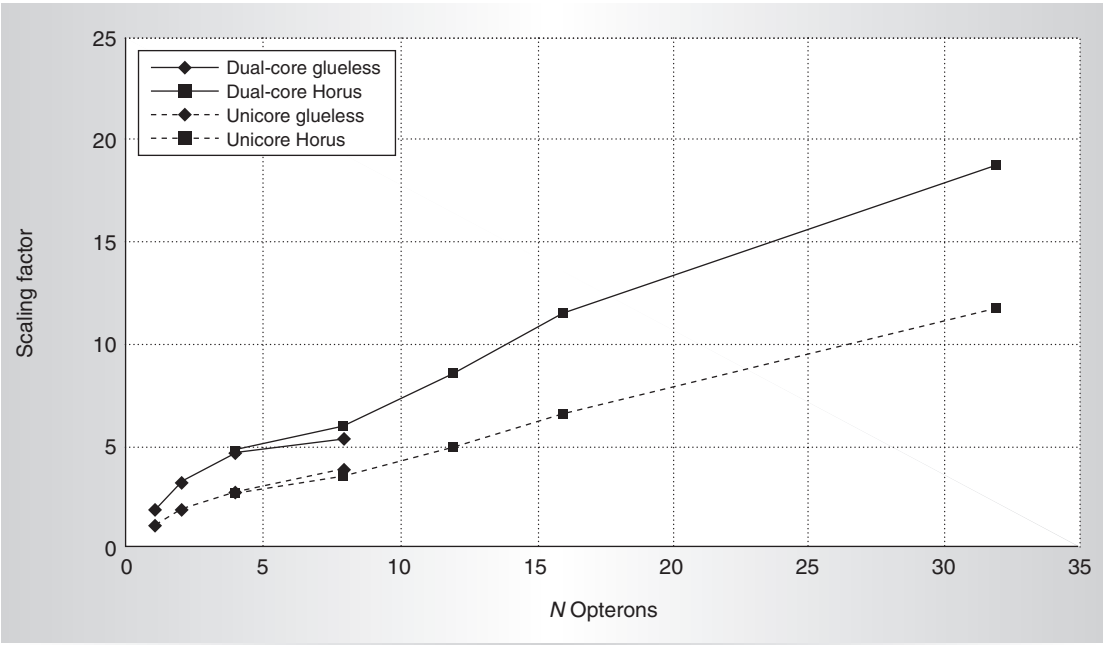

Figure 10. Estimated performance projections for an Opteron-only (glueless dual-core and unicore) versus Opteron plus Horus system (Horus dual core and unicore) running an OLTP application.

can recover from all soft errors without any help, including those caused by disparity; out-of-band signals; loss of signal; first-in, first-out overflows on the physical layer; cyclic redundancy check mismatches; packet loss; packet sequence ID mismatches; and illegal packets. The remote links will also recover from all soft errors.

All arrays (on and off chip) have error checking and correction that supports single-bit error correction, double-bit error detection, and scrubbing. Single-bit errors on a tag-array read will dynamically stretch the PE pipeline to allow for single-bit error correction until the pipeline is idle; when idle, the pipeline will shrink to a regular flow.

Horus has a JTAG (IEEE Std. 1149.1-1990) mailbox interface, which the Newisys service processor can use to periodically monitor Horus's health. The service processor can disable functionality in the case of a failure within Horus dynamically so that the system can make forward progress at a reduced performance. The JTAG mailbox provides side-

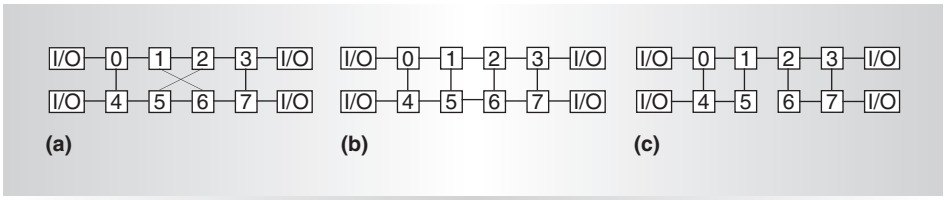

Figure 11. Configurations for an eight-way glueless SMP system: twisted (a) and simple (b) ladders, and dumbbell (c).

band access to all configuration, performance, and debug registers in Horus.

Performance results

Table 1 shows the latencies for different types of transactions in a system that uses Horus. These latencies are from the Opterons' L2 caches and include latencies inside Opterons as well, not just pin-to-pin latencies from Horus to the Opterons. These latencies are for the complete transaction up to the first critical data word.

As Table 1 shows, Horus has an appreciable latency reduction for transactions that exploit the Directory or RDC. For transactions to memory lines that hit in RDC, the transaction completes four times faster than a transaction not hitting in RDC. For transactions to memory lines that miss in Directory (the memory line is not cached in remote quads), the transaction completes three times faster than without Directory. Together, RDC and Directory significantly reduce the average transaction latency.

Figure 10 shows the performance scaling of Horus for an OLTP application. We used a cycle-accurate performance model and a short sample of traces for TPC-C benchmarks (version 5.1) at steady state. We assumed that Opterons were running at 2.8 GHz and had a 400-MHz dual-data-rate (DDR) DRAM and a 1-GHz HT link. The Horus core ran at 500 MHz, with off-chip RDC memory implemented using a 250-MHz DDR-FCRAM. The estimated hit rate for RDC and Directory was 90 percent.

Because of its multiple interfaces, multiple PEs, nonblocking crossbars, dual ports to off-chip memory, 32-entry-deep pending buffers in each PE, and large number of virtual channel credits in each link, Horus can handle a large number of transactions simultaneously. As the figure shows, increasing the number of outstanding transactions and CPU cores in the Opterons decreases the latency added because of Horus. Thus, the scaling of Horus improves significantly. Our other performance models support this.

Figure 11 shows three possible configurations for an eight-way glueless SMP system. The performance scaling in Figure 10 is with a simple ladder configuration, which relative to the twisted ladder, is easier to build and is modular. On the other hand, the simple ladder has a higher HT link usage than the twisted ladder and so degrades performance.

Removing one of the HT links from a simple ladder creates a dumbbell configuration, which scales worse (3.0 for single core and 4.0 for dual core) than a four-way glueless Opteron system because of the excessive traffic on the one HT link connecting the two quads.

Thus, although multiple eight-way configurations are possible, because some links must serve as dedicated I/O, there are fewer links available for constructing the interconnect. Further, routing constraints actually prevent minimum hops in what are otherwise optimal topologies. Our performance studies indicate that an eight-way twisted ladder offers the best trade-off between routing efficiency and I/O bandwidth, although building a twisted-ladder system is expensive, since it requires crossing four HT links at the center of the system.

With a twisted-ladder configuration, an eight-way glueless Opteron scales to 4.1 for unicore Opterons and to 6.7 for dual-core versions.

Since the tape-out of Horus in August 2004, we have been working on ramping up the chip and its surrounding infrastructure. We are simultaneously working on the design of the next-generation Horus, with the aim of improving its performance with future Opteron generations. MICRO

References

1. R. Oehler and R. Kota, "Horus: Large-Scale SMP for Opteron," 2004; http://mywebpages.comcast.net/davewang202/newisys/HorusHotChips2004.pdf.
2. A. Ahmed et al., "AMD Opteron Shared-Memory MP Systems," 2002; http://www.hotchips.org/archive/hc14/program/28_AMD_Hammer_MP_HC_v8.pdf.
3. N. Chetana et al., "The AMD Opteron Processor for Multiprocessor Servers," *IEEE Micro*, vol. 23, no. 2, Mar.-Apr. 2003, pp. 66-76.
4. Hypertransport Technology Consortium, "HyperTransport I/O Link Specification, rev. 1.10," Aug. 2003; http://www.hypertransport.org.
5. InfiniBand Specification 1.0.a, InfiniBand Trade Assoc., 2001; http://www.infinibandta.org/specs.

Rajesh Kota is a Horus architect at Newisys. His research interests include system engineering, microprocessor architectures, and low-power design. Kota has a BE in electronics and communication engineering from Osmania University, India, and an MS in computer engineering from the University of Maryland.

Rich Oehler has been chief technology officer at Newisys since its founding in 2000. His research interests include processor, system, and I/O architectures; operating systems; and compilers. He was a major contributor to the IBM 801, one of the first RISC machines; the IBM RS/6000 and PowerPC; and the IBM Summit chipset. Oehler has a BA in mathematics from St. Johns University.

Direct questions and comments about this article to Rajesh Kota or Rich Oehler, 10814 Jollyville Rd., Bldg. 4, Suite 300, Austin, TX 78759; {rajesh.kota, rich.oehler}@newisys.com.