On-Chip Interconnection Networks of the TRIPS Chip

Paul Gratz
The University of Texas at Austin

Changkyu Kim
Intel

Karthikeyan Sankaralingam
University of Wisconsin—Madison

Heather Hanson
IBM

Premkishore Shivakumar
Stephen W. Keckler
Doug Burger
The University of Texas at Austin

THE TRIPS CHIP PROTOTYPES TWO NETWORKS ON CHIP TO DEMONSTRATE THE VIABILITY OF A ROUTED INTERCONNECTION FABRIC FOR MEMORY AND OPERAND TRAFFIC. IN A 170-MILLION-TRANSISTOR CUSTOM ASIC CHIP, THESE NOCS PROVIDE SYSTEM PERFORMANCE WITHIN 28 PERCENT OF IDEAL NONCONTENDED NETWORKS AT A COST OF 20 PERCENT OF THE DIE AREA. OUR EXPERIENCE SHOWS THAT NOCS ARE AREA- AND COMPLEXITY-EFFICIENT MEANS OF PROVIDING HIGH-BANDWIDTH, LOW-LATENCY ON-CHIP COMMUNICATION.

Moore's law has recently led to architectures that incorporate multiple communicating components, such as chip-multiprocessor (CMP) and system-on-chip (SoC) designs. In the past, designs used ad hoc interconnect and bus protocols to convey control and data within a chip, but scaling these interconnects with frequency and greater numbers of components has motivated a shift toward networks on chip (NoCs). To date, however, relatively few such networks have been implemented and scaled to more than a handful of components.

For the Tera-ops, Reliable, Intelligently-adaptive Processing System—the TRIPS chip—we have designed and implemented two NoCs to support memory and operand traffic. The TRIPS on-chip network (OCN) replaces a traditional memory bus, and the operand network (OPN) replaces a traditional operand bypass and level-1 cache buses. Each of these networks is customized for its specific design constraints and traffic characteristics.

Figure 1 shows a die photograph of the TRIPS processor chip, along with a high-level block diagram of its components and the interconnection networks that connect those components. The block diagram shows the OCN connecting the two processor cores to the level-2 cache banks and various I/O units, including a chip-to-chip network interface unit that extends the OCN off chip for larger multiprocessor systems. Within each processor core, the OPN connects the different tiles that implement the register file, data caches, and execution units in a distributed processor microarchitecture. Because of their different purposes, the two networks differ in packet length and width, dimensions, and router design. Customizing the network design to the usage model and expected load is necessary for resource-efficient yet high-performance networks. This article examines the design, implementation, and behavior of these NoCs in a 170-million-transistor custom ASIC chip and discusses the insights gained from the network designs.

TRIPS processor overview

TRIPS is a processing system designed to achieve high performance through data-, instruction-, and thread-level parallelism. To provide the necessary execution and

0272-1732/07/$20.00 © 2007 IEEE Published by the IEEE Computer Society.

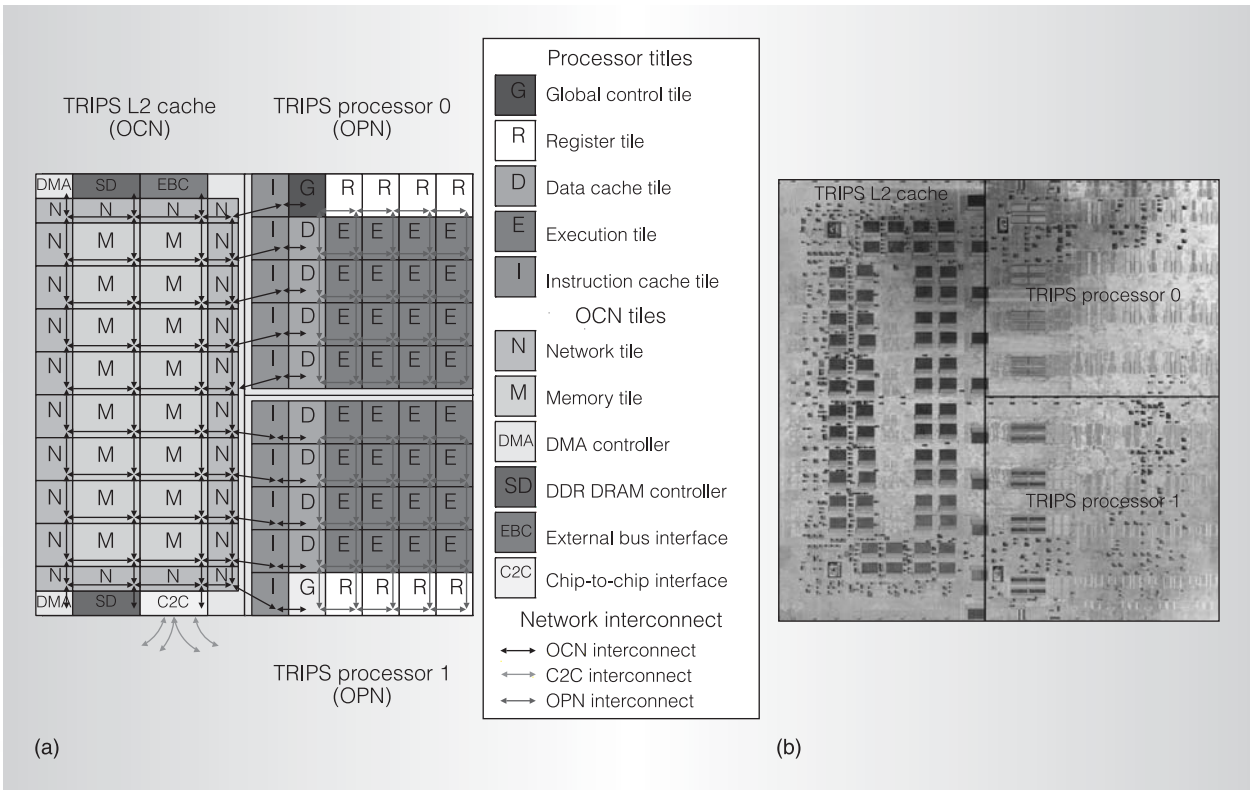

Figure 1. Block diagram of the TRIPS (Tera-ops, Reliable, Intelligently-adaptive Processing System) chip (a), and TRIPS prototype die photo (b).

memory resources (16 integer and floating-point units per processor), TRIPS is distributed into autonomous tiles connected via control and data networks. The TRIPS chip consists of 106 instances of 11 different tiles, each of which communicates directly with only its four nearest neighbors.

Figure 1a shows a tile-level diagram of the TRIPS chip with two processors, each with its own OPN links, and the L2 cache with its OCN links. The L2 cache consists of 24 network tiles (labeled N in the figure) to provide translation from system addresses to network addresses; and 16 memory tiles (labeled M), each containing 64 Kbytes of storage and an OCN router. The OCN interconnects these tiles to one another and to the processors, naturally pipelining many more requests and replies than traditional buses. The OCN has five ports connecting each processor core to the L2 cache, providing far higher cache bandwidth than a typical CMP L2 cache interconnect. The OCN architecture and protocols easily scale to more memory tiles for a larger L2 cache, or more on-die processors. The chip-to-chip network interface takes advantage of this scalability to allow interconnection of up to 64 TRIPS chips in a single, shared-memory, TRIPS system. Scaling beyond 64 chips is possible with minor changes to the system address map.

Each TRIPS processor core contains five types of tiles: 16 execution tiles (labeled E in Figure 1), which contain ALUs and reservation stations; four register tiles (labeled R), each containing a fraction of the processor register file; four data tiles (labeled D), each containing a L1 data cache bank; five instruction tiles (labeled I), each containing an L1 instruction cache bank; and one global control tile (labeled G), which orchestrates instruction fetch, execution, and commit.

These tiles are interconnected through the OPN for operand passing between the execution units and the primary memory system. A traditional broadcast bypass

network would not be feasible for the TRIPS design because bypass bus complexity and area are proportional to the square of the number of execution units. The scalability of the OPN enables the processor core to include the many execution and memory resources needed to exploit fine-grained concurrency. In addition to the OPN, the processor contains several control networks for implementing distributed protocols for instruction fetch, completion, and commit. Tiles communicate directly with only their nearest neighbors to keep wires short and mitigate the growing effects of global wire delays. Messages between distant tiles traverse the network routers in multiple hops.

ISA and execution model

TRIPS has an instruction set architecture with two key features: First, the hardware fetches, executes, and commits instruction blocks, rather than individual instructions, in an atomic fashion. Second, within a block, instructions send their results directly to other instructions waiting to execute, rather than communicating through a common register file.[1,2] The compiler constructs the blocks, which can contain up to 128 instructions. Because basic blocks typically contain only a handful of instructions, the TRIPS compiler uses techniques such as predication, loop unrolling, and function inlining to create large hyperblocks. After hyperblock formation, a scheduler maps the hyperblock onto the fixed array of 16 execution units, with up to eight instructions per execution tile. The scheduler attempts to minimize the distance between dependent instructions along the program's critical path while accounting for potential network contention.

Block execution

During block execution, the TRIPS OPN routes operands among the tiles according to the instruction placement. The TRIPS instruction formats contain target fields identifying the consumer instruction of the value produced. The hardware resolves those targets into coordinates for network routing. Operands passed between instructions on different tiles must traverse a portion of the OPN. The TRIPS execution model is inherently dynamic and data driven; operand arrival drives instruction execution, even if operands are delayed by unexpected or unknown memory latencies. The OPN must dynamically route operands across the processor due to the data-driven nature of execution.

Network implementations

The TRIPS design demands a high-bandwidth, low-latency interconnect to achieve system performance goals. To this end, the TRIPS prototype chip contains two data networks, the OCN and the OPN. Similar to arrangements in CMP and SoC designs, the OCN attaches two processors to L2 cache and IP block tiles along the top and bottom. The OPN network replaces bypass networks found in traditional processor designs. The ''Related work'' sidebar describes other approaches to NoCs.

On-chip network

The TRIPS OCN connects the two TRIPS processor cores to the individual banks that form the second-level cache and various IP blocks and I/O units. A primary requirement of the OCN is low latency, implying a single-cycle-per-hop router. Another requirement is network-addressing flexibility, so that cache banks can be used in multiple modes, including aggregating multiple banks to form L2 cache or scratch-pad memory. The mechanism for remapping system memory addresses to different OCN destination network addresses is a memory-mapped network address translation table. Any system address normally cached by the L2 banks can map to any memory tile determined by programming the address translation tables in the network tiles.

OCN implementation. Figure 1a shows a high-level block diagram of the TRIPS chip with two processors on the right, and the L2 cache with I/O units interconnected by the OCN on the left. The OCN consists of 16 memory tiles, each containing an OCN router and an L2 cache bank, and 24 network tiles containing an OCN router and the system address translation tables.

Related work

Although NoC architectures clearly have their roots in multicomputer interconnection networks, the substantial wire density available on chip changes the constraints on the network design.[1] Building on this observation, Dally and Towles claimed that replacing global wiring with a routed and modular on-chip interconnection network would shorten the design time and reduce wire-routing complexity.[2] The TRIPS architecture validates their conjecture, as TRIPS global routing and timing optimization were relatively simple, despite the chip's size and complexity.

On-chip routed networks are emerging in different multicore chip designs. The Massachusetts Institute of Technology Raw processor contains a statically routed, 4 × 4 mesh network to interconnect its processor tiles.[3] The Raw network is principally designed for the transmission of scalar operands, although it also transmits memory system traffic. More conventional multicore systems for commercial use employ between four and eight processors, and still use crossbars for interconnection.[4-6] As core counts increase, such designs will likely migrate to routed networks. For example, Intel's announced terascale prototype chip employs an 8 × 10 2D mesh network to interconnect its 80 processor cores.[7]

On-chip networks can also be found in growing numbers of SoC designs. The Cell Broadband Engine uses a concentric ring interconnection network to connect nine processing elements with external I/O and DRAM memory.[8,9] Other SoCs employ networks to connect large numbers of computation and memory elements.[10,11]

References

1. W. Dally and B. Towles, *Principles and Practices of Interconnection Networks,* 1st ed., Morgan Kaufmann, 2004.
2. W.J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," *Proc. 38th Design Automation Conf.* (DAC 01), ACM Press, 2001, pp. 684-689.
3. M.B. Taylor et al., "Scalar Operand Networks: On-Chip Interconnect for ILP in Partitioned Architecture," *Proc. 9th Int'l Symp. High-Performance Computer Architecture* (HPCA 03), IEEE CS Press, 2003, pp. 341-353.
4. U. Nawathe et al., "An 8-Core 64-Thread 64b Power-Efficient SPARC SoC," *Proc. IEEE Int'l Solid-State Circuits Conf.* (ISSCC 07), IEEE Press, 2007, pp. 108-109, 590.
5. "Quad Core: By Popular Demand," http://www.intel.com/quad-core.
6. J. Dorsey et al., "An Integrated Quad-Core Opteron Processor," *Proc. IEEE Int'l Solid-State Circuits Conf.* (ISSCC 07), IEEE Press, 2007, pp. 102-103.
7. S. Vangal et al., "An 80-Tile 1.28 TFLOPS Network-on-Chip in 65nm CMOS," *Proc. IEEE Int'l Solid-State Circuits Conf.* (ISSCC 07), IEEE Press, 2007, pp. 98-99, 589.
8. D. Pham et al., "Overview of the Architecture, Circuit Design, and Physical Implementation of a First-Generation Cell Processor," *IEEE J. Solid-State Circuits,* vol. 41, no. 1, Jan. 2006, pp. 179-196.
9. T. Ainsworth and T. Pinkston, "On Characterizing Performance of the Cell Broadband Engine Element Interconnect Bus," *Proc. 1st ACM/IEEE Int'l Symp. Networks-on-Chip* (NOCS 07), IEEE CS Press, 2007, pp. 18-29.
10. B. Khailany et al., "A Programmable 512GOPS Stream Processor for Signal, Image, and Video Processing," *Proc. IEEE Int'l Solid-State Circuits Conf.* (ISSCC 07), IEEE Press, 2007, pp. 272-273, 602.
11. A. Abbo et al., "XETAL-II: A 107GOPS, 600mW Massively-Parallel Processor for Video Scene Analysis," *Proc. IEEE Int'l Solid-State Circuits Conf.* (ISSCC 07), IEEE Press, 2007, pp. 270-271, 602.

Connected to the OCN along the top and the bottom are the I/O tiles, including two DMA controllers, two SDRAM controllers (labeled SD), the external bus controller (EBC), and the chip-to-chip (C2C) network controller. The C2C controller extends the OCN network protocols off chip, allowing up to 64 TRIPS chips to be connected directly together into a single, 128-processor, shared-memory system. This extensibility enables the scaling of the OCN to more L2 cache bank tiles and processors on die with little or no modification, as process technologies shrink.

Figure 2a shows the microarchitecture of an OCN router, including the translation tables seen in a network tile. To achieve low end-to-end packet latency, the OCN router has only one pipeline stage. The OCN router is otherwise typical of wormhole router designs. The router implements four packet classes, as required for protocol deadlock avoidance, with four dedicated virtual channels. The router contains enough incoming packet storage for two flow-control digits (flits) of data per direction, in each virtual channel. Incoming packets are latched into one of the input FIFO buffers in one of five input directions: north, south, east, west, or local for the L2 bank itself. A 6 × 6 crossbar network connects each input to every other possible output, including a loopback data path (labeled Config in Figure 2), which provides a mechanism for updating the translation tables.

Table 1 outlines the OCN network's design characteristics. The 2D mesh topology is a good fit for the 2D on-die substrate: All wires are point-to-point and short, and the router radix is low, minimizing complexity. Both the shorter wires and low radix help improve cycle time. To account for the asymmetric aspect ratio, we chose *Y-X* dimension order routing instead of *X-Y*, spreading out requests relative to the larger replies. The OCN's channel width is 138 bits, with 10 bits used for flow control and the remaining 128 bits for payload.

OCN design trade-offs. One trade-off involved the choice of flow-control method. Although virtual-channel routers, commonly seen off chip, exhibit less contention than

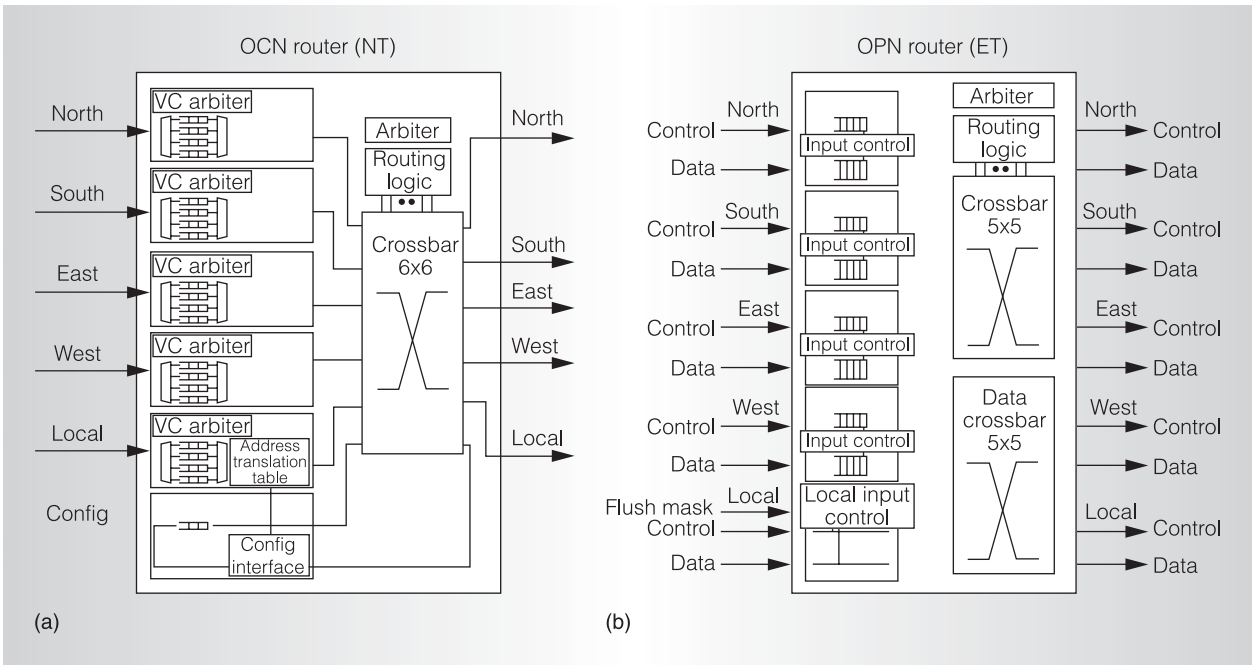

Figure 2. Block diagram of the on-chip network (OCN) router from a network tile (a) and the operand network (OPN) router from an execution tile (b). (NT: network tile; ET: execution tile.)

wormhole flow-control routers, we use wormhole routers. Wormhole routers reduce the area and timing constraints because the FIFO buffers in the OCN are already at the minimum size to account for the credit turnaround time. Adding extra virtual channels would significantly increase the router's area. Virtual-channel routing also adds more arbitration to the router's critical path, potentially affecting the cycle time or pipeline depth. Speculation could mitigate some of the effect on the critical path at the cost of higher design complexity.

The channel width decision also involved a trade-off. The OCN is primarily designed for cache line fills and spills, which in the TRIPS chip are 64 bytes long. We chose to make the payload of the OCN interface 16

Table 1. Design characteristics of OCN and OPN networks.

Characteristic	OCN	OPN
Topology	4 × 10 2D mesh	5 × 5 2D mesh
Channel width (payload + overhead)	128 + 10 = 138 bits	138 + 4 = 142 bits
Packet length	1 to 5 flits	1 flit
Packet classes	4	1
Input FIFO buffer depth	2	4
Routing algorithm	*Y-X* dimension order	*Y-X* dimension order
Flow control	Wormhole	N/A (single-flit packet)
Buffer management	Credit based	On-off based
Peak injection bandwidth	153 Gbytes/s	149 Gbytes/s
Bisection bandwidth	44 Gbytes/s	60 Gbytes/s
Router area	1.10 mm^2	0.43 mm^2
Network area (% of chip)	11%	8%
Router latency	1 cycle	1 cycle

bytes wide to balance between the router buffer area, efficiency of utilization in the network's input FIFO buffers, and the transmission latency of full cache lines. Because cache fill requests and spill acknowledgment replies require approximately 12 bytes and comprise one half of all packets, we chose a channel width that was the closest even divisor of a cache line size that would allow the small packets to fit in one flit.

Operand network

The OPN delivers operands between instructions on different execution tiles with a latency of one cycle per hop. Tight integration of the network into the processor core is necessary to achieve this low-latency network interface. In addition, two primary aspects of the TRIPS processor architecture simplify the router design, reducing routing latency. First, the block execution model preallocates reservation stations for all operand network packets, guaranteeing consumption of all OPN messages. Second, all OPN messages are a fixed 138 bits in length.

OPN implementation. Each 138-bit OPN message consists of a 29-bit control physical digit (phit) and a 109-bit data phit, where a phit is the amount of data processed by the OPN router in a cycle. The control phit encodes OPN source and destination node coordinates, along with identifiers to indicate which instruction to select and wake up in the target execution tile. The data phit includes a 64-bit data operand, an optional 40-bit address for store operations, and 5 bits for status flags. The OPN uses a total of 4 bits for flow control, making the entire channel 142 bits wide. The control phit leads the data phit by one cycle in the network to enable early wake-up of target instructions. Because the OPN supports different physical wires for the control and data phit, each OPN message consists of one flit, split into a control phit and a data phit.

Figure 2b shows a high-level block diagram of the OPN router. The OPN router has five inputs and five outputs, one for each ordinal direction (N, S, E, and W) and one for the local tile's input and output. Each ordinal-direction input has two four-entry-deep FIFO buffers, one for control phits and one for data phits. The local input has no FIFO buffer, which eliminates a cycle from the insertion of packets into the network. The OCN packet's control and data phits have separate 5×5 crossbars. All arbitration and routing occur on the control phit, in round-robin fashion among all incoming directions.

Table 1 details the OPN's design characteristics. Due to *Y-X* dimension-order routing and the consumption guarantee of messages, the OPN is deadlock free without requiring packet-class virtual channels. Flit-based flow control is unnecessary, because each packet is only one flit in length. The absence of virtual channels and flow control reduces arbitration delay and speeds routing, enabling single-cycle-per-hop routing in the OPN.

OPN-processor integration. The OPN creates a direct data path connection between the ALUs in adjacent execution tiles. In the source execution tile, the instruction selection logic is connected directly to the OPN's control-phit local input port, and the ALU's output latch is connected directly to the OPN's data-phit local input port. In the destination execution tile, the instruction wake-up logic is connected to the OPN's control-phit local output, and the local bypass network is connected to the OPN's data-phit local output. To minimize latency, early wake-up overlaps instruction pipeline control with operand data delivery, reducing the remote bypass time from two cycles to one, and improving performance by approximately 11 percent relative to a design where the wake-up occurs when the data arrives.[3] In addition, the separation of the control and data phits into separate physical networks with shared routing eliminates arbitration for the data phit and reduces network contention relative to a network that sends the header and payload on the same wires in successive cycles. The high wire density available on-chip makes this optimization inexpensive compared to off-chip networks.

Table 2. Critical path for OCN and OPN routers.

Component	Latency (ps)	Portion of clock cycle (%)
OCN critical path		
Source MT virtual channel arbitration	600	22
Source MT crossbar arbitration	550	20
Source MT output multiplex	90	3
Destination MT input FIFO buffer	590	22
Latch setup + clock skew	370	14
Total	2,200	81
OPN critical path		
Control phit generation	910	33
Source ET crossbar arbitration	420	15
Source ET output mux	90	3
Destination ET input FIFO buffer	710	26
Latch setup + clock skew	200	7
Total	2,330	85

MT: memory tile; ET: execution tile

The OPN is designed to carry short messages at a high rate. Approximately 90 percent of the traffic can fit within a packet payload of 99 bits, while the remaining 10 percent of traffic consists of L1 cache store requests and requires a packet payload of 138 bits. We chose a flit size of 138 bits for all flits because the injection rates in the OPN are high enough that reducing the network occupancy reduces the contention of the network significantly, relative to designs in which 10 percent of packets are two flits. Also, single-flit packets reduced router complexity and helped achieve the desired clock frequency of 366 MHz.

Network comparison

The TRIPS chip networks have different functions within the larger TRIPS system. Although we could have used a single, generalized network to interconnect all the tiles, customizing the networks to fit their function provides higher system performance at a lower cost in terms of area.

Area. The TRIPS processor is manufactured using 130-nm IBM ASIC technology. In both the OPN and OCN routers, the router input FIFO buffers dominate the routers, consuming 75 percent of the router area. Table 1 shows the area consumed by each router and the percentage of the chip area that each network consumes. The OCN routers are more than twice the size of the OPN routers, primarily because of the extra buffering required for virtual channels. The networks' overall area consumption was higher than initially projected, primarily because of the wide input FIFO buffers required by the wide channel widths. For both networks, we sacrificed die area to gain system performance. By comparison, a bus-based interconnect with a similar number of connections, many sending and receiving at the same time, would be infeasible.

Timing. Neither the OCN nor the OPN contained the chip's clock-limiting critical path, although the OPN's critical path consumes 85 percent of the clock period. Further pipelining would not affect the 2.73-ns clock period (at 366 MHz) of the TRIPS chip. We found that wire delay was small in our 130-nm process, given the relatively short transmission distances. Balancing router delay and wire delay, however, will be more challenging in future process technologies, and will increase the need for faster routers.

Table 2 shows the delay for the different elements of the OCN and OPN critical paths. Despite larger and more complex routers, the OCN path is less critical than

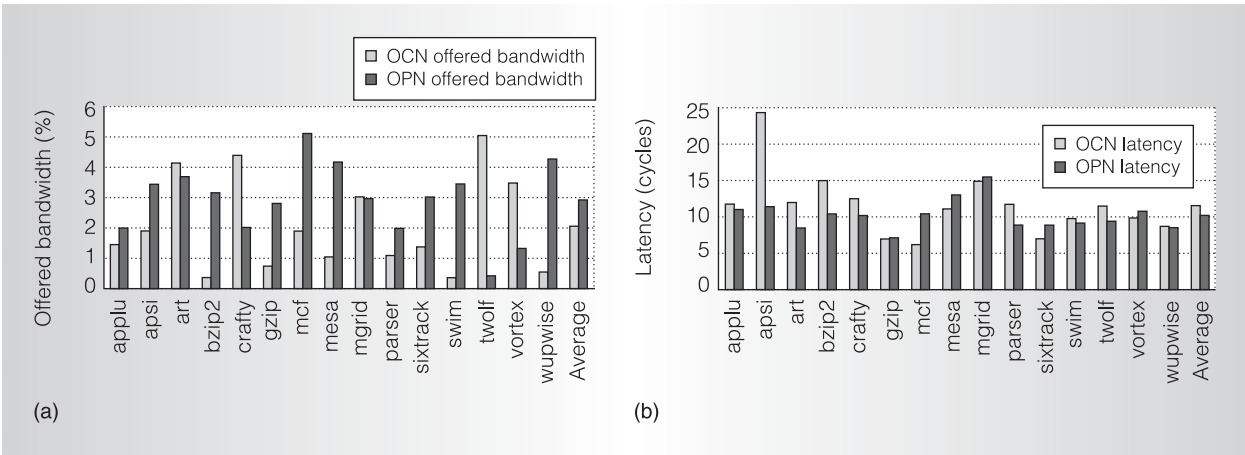

Figure 3. OCN and OPN offered rate (a) and latency (b) for SPEC CPU2000 benchmarks.

the OPN, primarily because the OPN lacks input FIFO buffers on the local port. In the OCN, it would be impossible to inject packets without an input FIFO buffer, because of the higher complexity associated with the packet-class virtual channels. The delay associated with creating the control phit in the OPN path exceeded our expectations. If necessary, this path could be improved by generating the control phit at instruction fetch time and storing it until use.

Network evaluations

We evaluated the OPN and OCN to characterize the message workload, and examined the sensitivity of program performance to network contention, to determine the efficiency and efficacy of the networks' design.

Methodology

For long-running benchmarks, we obtained workload traces from a TRIPS processor core simulator. These traces were processed by a stand-alone network simulator that can be configured with either the OCN or OPN design parameters to calculate bandwidth and latency. For more detailed analysis, we also used a low-level simulator that accurately models all aspects of a TRIPS chip, including OCN and OPN network contention. Our experiments included 16 of the SPEC CPU2000 benchmarks, run with the Minne-SPEC reduced input set.[4] However, even the Minne-SPEC versions of SPEC CPU2000 were too long-running for full system simulation, so we used checkpoints to approximate the full application behavior.[5]

Traffic comparison

Figure 3 shows the offered bandwidth as a percentage of peak injection bandwidth, and the average packet latency in cycles for the OCN and OPN traffic of selected SPEC CPU2000 benchmark traces on the network simulator. On average, the offered rate for OCN traffic is 29 percent lower than that of the OPN. The latency of OCN packets average 11.5 cycles, compared to 10.2 cycles in the OPN. The average hop count for traffic in the OCN is approximately 5 hops, versus approximately 2 hops in the OPN, so a greater portion of the latency in the OPN is due to contention. The offered rates in the OCN and OPN are fairly low relative to the latencies shown.

Uniform traffic, with these offered rates, would be well below the saturation point for mesh networks like the OCN and OPN. Our previous work has shown that the traffic offered by real applications, such as SPEC CPU2000 benchmarks, is nonuniform in space (for the OPN) and injection time (for the OCN).[3,6] The complier mediates between placing dependent instructions further apart (which load-balances the network and reduces contention) versus placing instructions closer together (which decreases the distance that operands on the critical path must travel to address

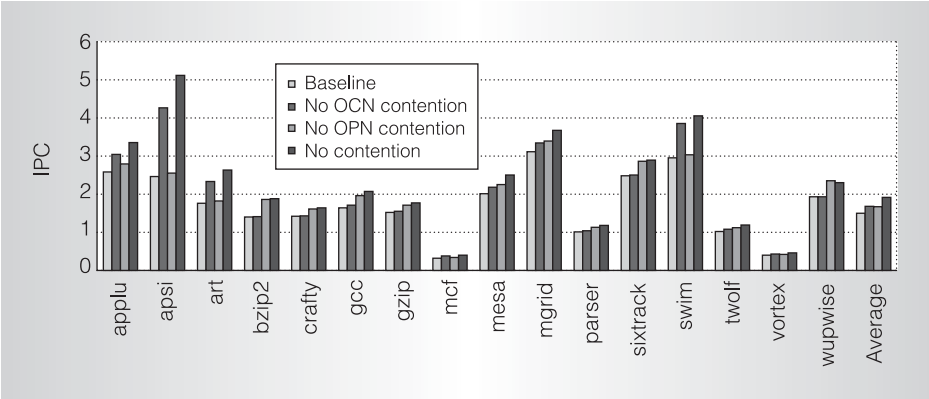

Figure 4. Effect of OCN and OPN contention on TRIPS instructions per cycle for SPEC CPU2000 benchmarks.

the nonuniformity of requests in space). Head-of-line blocking, in which contended packets at the front of a queue block the forward progress of packets behind them, also causes contention in the OPN, and to a lesser degree in the OCN. Virtual-channel flow control could mitigate this, but at the cost of increasing buffer area and cycle time or pipeline depth.

System performance sensitivity to network contention

Figure 4 shows the TRIPS chip's overall performance, measured in average instructions per cycle (IPC). For each of the subset of the SPEC CPU2000 benchmarks, Figure 4 shows four IPC numbers: the baseline system as implemented in the TRIPS chip, the same system with no contention modeling in the OCN, with no contention modeling in the OPN, and finally with no contention modeling in either network.

The benchmarks fall into two broad categories: those that are highly sensitive to OCN contention (applu, apsi, art, and swim) and those that are not. Benchmarks sensitive to OCN contention contain far more L2 cache traffic relative to the others. Nearly all benchmarks see a moderate performance decrease from OPN contention, with a geometric mean 12 percent higher than baseline. The results for wupwise show a 2 percent performance drop between OCN contention without OPN contention and fully noncontended. This anomaly is caused by a reordering of L1 data cache fills leading to slightly more L1 data cache conflict misses in the noncontended case. The comparison of baseline to no contention yields an average of 28 percent improvement in performance across all benchmarks. Contention, therefore, is still a contributor to latency; further network optimization could recover more performance.

NoCs are emerging as critical components of chip-level processor, memory, and system architectures. The TRIPS implementation of the OCN and OPN demonstrate that NoCs can be constructed to deliver high bandwidth and low latency without adversely affecting timing and area. Our TRIPS network performance evaluation indicates that further research is needed to reduce contention through better exploitation of on-chip wire densities or by employing some form of adaptive routing. However, we expect that designing more aggressive congestion control while maintaining both high clock frequency and low-latency routers will be challenging. MICRO

References

1. D. Burger et al., ''Scaling to the End of Silicon with EDGE Architectures,'' *Computer*, vol. 37, no. 7, July 2004, pp. 44-55.
2. K. Sankaralingam et al., ''The Distributed Microarchitecture of the TRIPS Prototype Processor,'' *Proc. 39th ACM/IEEE Int'l Symp. Microarchitecture* (MICRO 06), IEEE CS Press, 2006, pp. 480-491.

3. P. Gratz et al., ''Implementation and Evaluation of a Dynamically Routed Processor Operand Network,'' *Proc. 1st Int'l Symp. Networks-on-Chip* (NOCS 07), IEEE CS Press, 2007, pp. 7-17.
4. A.J. KleinOsowski and D.J. Lilja, ''MinneSPEC: A New SPEC Benchmark Workload for Simulation Based Computer Architecture Research,'' *Computer Architecture Letters*, vol. 1, no. 1, Jan. 2002, p. 7.
5. T. Sherwood, E. Perelman, and B. Calder, ''Basic Block Distribution Analysis to Find Periodic Behavior and Simulation Points in Applications,'' *Proc. Int'l Symp. Parallel Architectures and Compilation Techniques* (PACT 01), IEEE CS Press, 2001, pp. 3-15.
6. P. Gratz et al., ''Implementation and Evaluation of On-Chip Network Architectures,'' *Proc. IEEE Int'l Conf. Computer Design* (ICCD 06), IEEE Press, 2006, http://www.iccd-conference.org/proceedings/2006/paper_174.pdf.

Paul Gratz is a PhD student in the Department of Electrical and Computer Engineering at the University of Texas at Austin. His research interests include on-chip interconnection networks, computer architecture, and distributed systems. Gratz has an MS in electrical engineering from the University of Florida. He is a member of the IEEE.

Changkyu Kim is a research scientist in Intel's Microprocessor Technology Lab. His research interests include high-performance microprocessors, memory systems, chip-multiprocessors, and on-chip networks. Kim has a PhD in computer sciences from the University of Texas at Austin.

Karthikeyan Sankaralingam is an assistant professor of computer science at the University of Wisconsin-Madison. His research interests include computer architecture, VLSI design, and adaptive systems. Sankaralingam has a PhD in computer sciences from the University of Texas at Austin. He is a member of the IEEE and the ACM.

Heather Hanson is a member of the Power-Aware Systems Department at IBM's Austin Research Laboratory. Her research interests include microprocessor design and strategies for power and thermal management. Hanson has a PhD in electrical and computer engineering from the University of Texas at Austin. She is a member of the IEEE.

Premkishore Shivakumar recently earned a PhD in computer science at the University of Texas at Austin. His research interests include high-performance computer architecture, subject to the constraints imposed by performance, power consumption, and reliability. He is a member of the IEEE and the ACM.

Stephen W. Keckler is an associate professor of both computer sciences and electrical and computer engineering at the University of Texas at Austin, where he co-leads the TRIPS project. His research interests include computer architecture, interconnection networks, and parallel processor architectures. Keckler has a PhD in electrical engineering and computer science from the Massachusetts Institute of Technology. He is a senior member of the IEEE and the ACM.

Doug Burger is an associate professor of both computer sciences and electrical and computer engineering at the University of Texas at Austin, where he co-leads the TRIPS project. His primary research interest is computer architecture. Burger has a PhD in Computer Science from the University of Wisconsin-Madison. He is a senior member of the IEEE and the ACM.

Direct questions and comments about this article to Paul Gratz, Dept. of Electrical and Computer Engineering, The University of Texas at Austin, 1 University Station C0803, Austin, TX 78712-10240; pgratz@cs.utexas.edu.