Power-driven Design of Router Microarchitectures in On-chip Networks

Hangsheng Wang Li-Shiuan Peh Sharad Malik

Department of Electrical Engineering,
Princeton University, Princeton, NJ 08544
{hangshen,peh,sharad}@ee.princeton.edu

Abstract

As demand for bandwidth increases in systems-on-a-chip and chip multiprocessors, networks are fast replacing buses and dedicated wires as the pervasive interconnect fabric for on-chip communication. The tight delay requirements faced by on-chip networks have resulted in prior microarchitectures being largely performance-driven. While performance is a critical metric, on-chip networks are also extremely power-constrained. In this paper, we investigate on-chip network microarchitectures from a power-driven perspective. We first analyze the power dissipation of existing network microarchitectures, highlighting insights that prompt us to devise several power-efficient network microarchitectures: segmented crossbar, cut-through crossbar and write-through buffer. We also study and uncover the power saving potential of an existing network architecture: express cube. These techniques are evaluated with synthetic as well as real chip multiprocessor traces, showing a reduction in network power of up to 44.9%, along with no degradation in network performance, and even improved latency-throughput in some cases.

1 Introduction

On-chip networks have been widely proposed as the interconnect fabric for high-performance systems-on-a-chip (SoCs) [3, 9, 15], and demonstrated in several chip multiprocessors (CMPs) [14, 20]. As these networks are facing tight delay requirements [21], prior designs and microarchitecture studies have been heavily performance-driven, aiming towards lowering network delay to that of pure wire transmission latency [13, 14, 20].

However, the targeted systems of on-chip networks are increasingly becoming power-constrained. Battery life is of primary concern in embedded SoCs in PDAs, laptops and other mobile devices. In enterprise and desktop environments, system power budget, pressured by cooling and packaging costs, is the key constraint faced by designers today in systems such as server blades, storage bricks and PCs. With the increasing demand for interconnect bandwidth, on-chip networks are taking up a substantial portion of system power budget, e.g. the MIT Raw on-chip network which connects 16 tiles of processing elements consumes 36% of total chip power, with each router dissipating 40% of individual tile power.

This drove us to rethink network microarchitecture design from a power-driven perspective and investigate what constitutes the ideal transmission energy in networks. To understand the power characteristics of prior performance-driven network microarchitectures, we modeled the on-chip networks of two CMPs – the MIT Raw [20] and the UT Austin TRIPS [14] and obtained their power profile by using Orion [24], a power-performance simulator for interconnection networks. Our modeling highlights the significant power dissipation of on-chip networks, and brings valuable insights on the relative power composition of different microarchitecture components in on-chip networks.

Based on our analysis and insights, we devise three new microarchitectures: segmented crossbar, cut-through crossbar and write-through buffer. We also study the power saving potential of an existing network architecture: express cube. These techniques target energy reduction of key router components towards the ideal network transmission energy. Each mechanism's impact on power, performance and area is first analyzed in-depth through power modeling and probabilistic analysis, then evaluated with a cycle-accurate network simulator. Our simulations show that these mechanisms can achieve significant power savings of 44.9% for synthetic uniform random traffic, and 37.9% for the TRIPS traffic traces, as compared to a baseline network configuration based on current on-chip network designs, with no performance degradation.

The rest of this paper is organized as follows: section 2 introduces the power characteristics of prior performance-driven on-chip network designs, along with a characterization of ideal network energy consumption, and the power profiles of the Raw and TRIPS on-chip networks. Section 3 describes four power-efficient network microarchitectures,

Proceedings of the 36th International Symposium on Microarchitecture (MICRO-36 2003)
0-7695-2043-X/03 $17.00 © 2003 IEEE

analyzing the power-performance-area impact of different points in the design space for each mechanism. Section 4 delves into our simulation results of the proposed mechanisms, evaluated against synthetic and real traffic traces. A discussion of prior related work follows in section 5, and section 6 concludes the paper.

2 Background and analysis

2.1 Background on on-chip networks

On-chip networks have been proposed for chip multiprocessors (CMPs) as well as heterogeneous systems-on-a-chip [15]. In this work, we focus on fine-grained CMPs [14, 20] that transport operands instead of more traditional multiprocessors [1, 2, 22] that carry cache lines or user messages, as such fine-grained CMPs with on-chip networks have already been demonstrated and fabricated, providing invaluable access to design details for power characterization and analysis.

In these fine-grained CMPs, whose networks are also named scalar operand networks [21], operand bypassing and forwarding delay between processing elements is absolutely critical to system performance, so network delay stands out as the most important performance metric. This has resulted in prior designs of on-chip network microarchitectures being heavily performance-driven, geared towards lowering network delay as much as possible.

In the drive towards lower network delay, on-chip networks tend to choose simple, fast routing protocols, such as dimension-ordered routing. For example, Raw's dynamic network uses a routing algorithm similar to dimension-ordered routing in that each flit[1] can turn at most once. On-chip networks also tend to opt for two-dimensional topologies such as meshes and tori. While high-dimensional topologies have shorter average hop count, they are less desirable for the small scale of on-chip networks of today that tend to range from 4×4 to 8×8 nodes. This is due to constraints on chip size and hop delay, as they lead to longer channels as well as uneven channel delay [6]. Both Raw and TRIPS use 2-D topologies.

2.2 Power analysis of on-chip networks

As characterized previously in [24], the energy consumed when transmitting a data flit[2] is:

$$\begin{aligned} E_{flit} &= (E_{wrt} + E_{rd} + E_{arb} + E_{xb} + E_{lnk}) \cdot H \\ &= (E_{buf} + E_{arb} + E_{xb} + E_{lnk}) \cdot H \end{aligned} \quad (1)$$

[1] A flit is short for flow control unit, a fixed-length segment of a packet.
[2] We use data flits in our discussion for simplicity.

where E_{wrt} is the average energy dissipated when writing a flit into the input buffer, E_{rd} is the average energy dissipated when reading a flit from the input buffer, $E_{buf} = E_{wrt} + E_{rd}$ is average buffer energy, E_{arb} is average arbitration energy, E_{xb} is average crossbar traversal energy, E_{lnk} is average link traversal energy and H is the number of hops traversed by this flit.

Equation (1) can be reformulated as:

$$E_{flit} = E_R \cdot H + E_{wire} \cdot D \quad (2)$$

where $E_R = E_{buf} + E_{arb} + E_{xb}$ is average router energy, E_{wire} is average link wire transmission energy per unit length assuming optimally-placed repeaters and D is the Manhattan distance between source and destination.

The ideal flit transmission energy is that dissipated by a dedicated wire linking source and destination, corresponding to the minimum physical activity required to complete the transmission:

$$E_{flit_{ideal}} = E_{wire} \cdot D \quad (3)$$

But in reality, more than ideal energy is consumed as the network shares and multiplexes links between multiple source-destination flows, thus requiring intermediate routers that dissipate additional energy E_R.

To understand the power profile of on-chip networks, we used Orion to model the power consumption of the Raw and TRIPS networks. Orion's power models have been validated against Raw to be within 10% error of circuit-level power estimations. Details of the power profiling can be found in [23], here we only give summaries:

Network power vs. total system power. In the Raw multiprocessor system, interconnection networks consume 7.1W power, which is 36% of total chip power, as estimated by the Raw designers using circuit-level tools [10]. This result highlights that networks consume a substantial portion of total system power in CMPs.

Network power composition. Table 1 summarizes the average power composition of the Raw and TRIPS data networks. These numbers will vary with different network parameters, but they clearly convey that unlike off-chip networks, where link power dominates, buffers, crossbars and links are equally important for reducing on-chip network power.

Table 1. Raw and TRIPS average network power composition.

	input buffer	crossbar	arbiter	link
Raw	31%	30%	~0%	39%
TRIPS	35%	33%	1%	31%

3 Power-efficient network microarchitectures

Our power analysis of on-chip networks as encapsulated in Equation (2) highlights several avenues that can be targeted for reducing flit transmission energy – router energy E_R, hop count H and wire transmission energy per unit length E_{wire}. E_R mainly consists of buffer energy and crossbar energy. Source-destination distance D is largely determined by physical placement and is considered an input to network microarchitecture design.

Many circuit techniques have been proposed for reducing E_{wire}, such as low-swing signaling [9, 25] and bus encoding schemes [19]. In this paper, we focus on the architecture level instead, devising, analyzing and evaluating four network microarchitectural/architectural techniques that target different components of E_R and H.

3.1 Segmented crossbar

Matrix crossbar is a common crossbar design. Figure 1(a) shows the schematic of a 4×4 matrix crossbar. Each line in the figure represents a flit-wide bus, with tri-state buffers at cross points, enabling connections from input ports to output ports. From the figure we can see that when a flit enters from input port E and leaves through output port W, the entire input lines and output lines switch. Useful switching, however, is indicated by the dotted line – less than half of the actual switching capacitance.

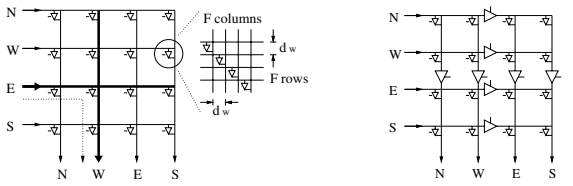
This prompted us to propose segmented crossbars, a simplified application of segmented buses [4]. Figure 1(b) shows its schematic. Each input/output line is evenly divided into M segments by tri-state buffers. Switch arbiters generate and configure the control signals of these tri-state buffers so that only the minimally needed wire segments switch, thus reducing E_{xb}.

Power analysis. Crossbar traversal energy E_{xb} can be formulated as:

$$E_{xb} = E_{xb\_in} + E_{xb\_out} + E_{xb\_ctr} \tag{4}$$

where E_{xb_in} is the energy dissipation of crossbar input lines, E_{xb_out} is the energy dissipation of crossbar output lines, and E_{xb_ctr} is the energy dissipation of control lines of tri-state buffers.

E_{xb_in} is determined by the input line capacitance C_{in}, switching activity A_{in}, and flit size F; while E_{xb_out} is similarly determined by the output line capacitance C_{out}, switching activity A_{out} and F. A segmented crossbar reduces C_{in} and C_{out}, and has no impact on switching activities A_{in} and A_{out}. As it requires more control lines for segment tri-state buffers, E_{xb_ctr} increases. The negative impact on power is minimal, given far fewer control lines than input/output lines unless M is larger than flit size. All these details are modeled by extending Orion's crossbar power model.



(a) A 4×4 matrix crossbar.

(b) A 4×4 segmented crossbar with 2 segments per line.

Figure 1. Schematic of a matrix crossbar and a segmented crossbar. F is flit size in bits, d_w is track width, E, W, N, S **are ports.**

We define C_{in} to be the input line capacitance of a normal, unsegmented matrix crossbar, C_{ti} to be the input capacitance of one tri-state buffer and C_{to} to be the output capacitance of one tri-state buffer. Assuming that traffic is evenly distributed among all input and output ports, the probability of the ith segment being traversed by incoming flits is:

$$P_i = \frac{M-i+1}{M}, \quad i \in [1, M]$$

and the switching capacitance of the ith segment is:

$$\begin{aligned} C_1 &= \frac{C_{in}}{M} + C_{ti} \\ C_i &= \frac{C_{in}}{M} + C_{ti} + C_{to}, \quad i = 2, \ldots, M-1 \\ C_M &= \frac{C_{in}}{M} + C_{to} \end{aligned}$$

So total effective switching capacitance is:

$$\begin{aligned} C'_{in} &= \sum_{i=1}^{M} P_i \cdot C_i \\ &= \frac{M+1}{2M} \cdot C_{in} + \frac{(M+2)(M-1)}{2M} \cdot C_{ti} + \\ &\quad \frac{M-1}{2} \cdot C_{to} \end{aligned} \tag{5}$$

The same analysis can be applied to C_{out}:

$$\begin{aligned} C'_{out} &= \frac{M+1}{2M} \cdot C_{out} + \frac{(M+2)(M-1)}{2M} \cdot C_{ti} + \\ &\quad \frac{M-1}{2} \cdot C_{to} \end{aligned} \tag{6}$$

From Equations (5) and (6), an upper bound of power savings with 2, 3 and 4 segments is 25%, 33.3% and 37.5% respectively, which is calculated by ignoring C_{ti}, C_{to} and

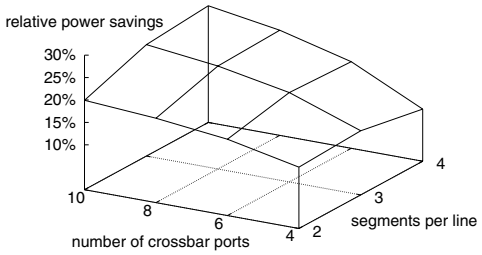
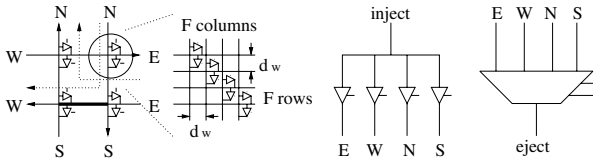


Figure 2. Segmented crossbar power savings (relative to matrix crossbar).



(a) A 4×4 cut-through crossbar.

(b) Extending to a 5×5 crossbar.

Figure 3. Schematic of cut-through crossbars. F is flit size, d_w is track width, E, W, N, S **are ports.**

control lines. As M increases, these ignored terms become more significant and the bound becomes looser. So we can conclude that adding more segments beyond 2 or 3 does not buy enough power savings to compensate for the overhead.

Figure 2 shows the power savings of several segmented crossbar configurations relative to matrix crossbar, estimated by our power models, assuming 64-bit flit and 0.1μm technology. Note that several configurations are not quite valid, e.g. 3 does not divide 4, but are shown to reflect general trends.

From the figure we can see that large-scale crossbars tend to gain higher power savings from segmentation since the overhead of tri-state buffers gets relatively smaller. On the other hand, for a fixed crossbar size, increasing the number of segments does not always result in higher power savings, e.g. a 3-segment 4×4 crossbar saves more power than a 4-segment 4×4 crossbar.

Impact on performance. A 2-segment segmented crossbar has negligible impact on performance since a single tri-state buffer can be easily absorbed into the existing repeater chains and will not affect the duration of the crossbar traversal pipeline stage. More segments may increase crossbar delay, but they do not necessarily save more power either.

Impact on area. Segmenting has no impact on area. Crossbar area is determined by the width and height of the matrix grid, and since cross point tri-state buffers and repeater chains can fit within the grid, so can segment tri-state buffers.

3.2 Cut-through crossbar

Cut-through crossbar is another microarchitectural mechanism we propose for reducing crossbar traversal energy E_{xb}. While segmented crossbars are motivated by the inherent operation of a crossbar, cut-through crossbars leverage insights on network traffic patterns and routing protocols in on-chip networks.

Figure 3(a) sketches a 4×4 cut-through crossbar, where each input port is directly connected to the output port of the opposite direction by a flit-wide bus, e.g. E→W, N→S, so only turns from one dimension to another go through tri-state buffers at cross points.

Essentially, cut-through crossbars optimize for the common case – straight-through traffic, so straight-through traffic incurs lower energy consumption as well as faster traversal. However, it does so at the expense of the connectivity of the crossbar switch.

A cut-through crossbar supports almost the same connectivity as the original matrix crossbar, with the exception of a few combinations of turns as they share common wires. One example is shown by the two dotted lines in Figure 3(a) – connections E→N and N→W conflict because they both use a common line segment (shown in bold). In this case, one flit needs to wait for the next crossbar traversal cycle. An observation is that any connectivity restriction induced by cut-through crossbars consists of at least two turns: one from the horizontal dimension to the vertical dimension, another from the vertical dimension to the horizontal dimension. Therefore, cut-through crossbars incur no performance penalty when a simple, dimension-ordered routing protocol is used, because with such protocols, flits can only turn from one dimension to another, but not vice versa, hence inherently preventing any conflict. The trend towards simplistic routing protocols in on-chip networks thus makes cut-through crossbar a feasible choice.

Cut-through crossbars target 4×4 fabrics, a common switch size in on-chip 2-D torus or mesh topologies. When injection port and ejection port are considered, since a cut-through crossbar is not easily scalable to support a 5×5 fabric, a 4:1 mux and a 1:4 demux are used in parallel with the crossbar, as shown in Figure 3(b). Each input port and output port is properly muxed or demuxed to prevent interference among the three components.

Power analysis. Recall Equation (4), $E_{xb} = E_{xb\_in} + E_{xb\_out} + E_{xb\_ctr}$. Cut-through crossbars reduce E_{xb_in} and

E_{xb_out} by reducing the input/output line length and capacitance. The input line capacitance is:

$$C_{in} = C_L + C_T$$

where C_L is wire capacitance and C_T is the capacitance of attached tri-state buffers. In Figure 1(a) each line represents an F-bit bus, where F is flit size. So a 4×4 matrix crossbar can be regarded as a $4F \times 4F$ grid, and the line length is:

$$L_M = 4F \cdot d_w$$

where d_w is track width. Similarly, a cut-through crossbar can be regarded as a $2F \times 2F$ grid, and its line length is:

$$L_C = 2F \cdot d_w$$

C_L is proportional to line length, and is in turn proportional to flit size. Assuming large enough flit size, hence $C_T \ll C_L$ and $C_{in} \approx C_L$, and since $L_C = \frac{1}{2}L_M$, E_{xb_in} of a cut-through crossbar is one half of that of a matrix crossbar. Similarly, we can derive that the average control line length is one half of that of a matrix crossbar, and so is E_{xb_ctr}. $E_{xb\_out} = 0$ in cut-through crossbars since each input line extends directly to its opposite output port and there is no separate output line.

Relative to matrix crossbar traversal energy, cut-through crossbar traversal energy is:

$$E'_{xb} = \frac{1}{2}E_{xb\_in} + \frac{1}{2}E_{xb\_ctr} \quad (7)$$

For a matrix crossbar with the same number of input ports and output ports, $E_{xb\_in} \approx E_{xb\_out}$. Also since there are far fewer control lines than input/output lines, which implies $E_{xb\_ctr} \ll E_{xb\_in}$, a cut-through crossbar consumes roughly $\frac{1}{4}$ energy of a matrix crossbar, and an upper bound of power saving is 75%.

We extended Orion's power models for cut-through crossbars and Table 2 shows the estimated power savings relative to matrix crossbar, for varying flit sizes and technologies, assuming 0.5 switching probability and 0.5 flit arrival rate.

Table 2. Cut-through crossbar power savings (relative to matrix crossbar).

flit size	64 bits	128 bits	256 bits
0.1μm	39.4%	47.2%	52.0%
0.18μm	33.6%	43.0%	50.0%

Since each line is connected with 4 tri-state buffers, whose capacitance contribution is fixed, and line length is proportional to flit size, larger flit size leads to greater significance of wire capacitance and higher power saving. The increase in power savings as technology scales down is due to the increasing impact of wire coupling capacitance.

Impact on performance. To get an upper bound of latency increase due to the connectivity restrictions of a cut-through crossbar, we assume uniform random traffic and a worst-case routing algorithm which gives each incoming flit the same probability of leaving from any other direction. This is a reasonable assumption because, although not impossible, it is unrealistic to design a routing algorithm which makes twice as many turns as straight traffic. Assume a 4×4 router and let λ denote the flit arrival rate at each input port. Then, for each input port, the probabilities of traffic turning and not turning at any cycle (assuming no u-turns) are:

$$\begin{aligned} p_{turn} &= \frac{2}{3}\lambda \\ p_{not\_turn} &= 1 - \frac{2}{3}\lambda \end{aligned}$$

With two input ports per dimension, the following equations derive the probabilities of each dimension having two turns, one and only one turn, and any one turn respectively:

$$\begin{aligned} P_{2\_turn} &= p_{turn} \cdot p_{turn} \\ P_{1\_turn} &= 2 \cdot p_{turn} \cdot p_{not\_turn} \\ P_{any\_turn} &= 1 - p_{not\_turn} \cdot p_{not\_turn} \end{aligned}$$

Now consider the conflicting condition which contributes additional delay, of which there are three scenarios:

- The horizontal dimension has two turns and the vertical dimension has turning traffic. They conflict as two turns from one dimension use up all wire resources of this dimension, preventing them from being used by turns from another dimension.
- The horizontal dimension has one turn and the vertical dimension has two turns.
- Both the horizontal dimension and the vertical dimension have exactly one turn. They conflict with probability p_{xy}. A simple enumeration reveals that $p_{xy} = \frac{3}{4}$.

So the conflicting probability (plotted in Figure 4) is:

$$\begin{aligned} P_C &= P_{2\_turn} \cdot P_{any\_turn} + P_{1\_turn} \cdot P_{2\_turn} + \\ &\quad P_{1\_turn} \cdot P_{1\_turn} \cdot p_{xy} \\ &= \left(\frac{2}{3}\lambda\right)^2 \cdot \left(\frac{3}{4} + \lambda - \frac{5}{9}\lambda^2\right) \end{aligned}$$

From this equation, we can see that $P_{Cmax} = P_C|_{\lambda=1} = \frac{43}{81}$, which means that at full load ($\lambda = 1$), the latency of the crossbar is increased by 53%. With a typical 3-stage router pipeline and a single cycle link propagation delay, a cut-through crossbar will introduce at most 20% additional delay, which is a favorable trade-off for energy-delay product, even with such an unrealistic worst-case routing algorithm.

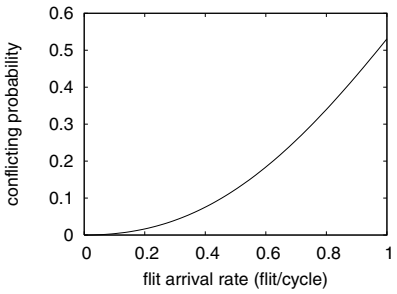


Figure 4. Conflict probability vs. flit arrival rate for a cut-through crossbar.

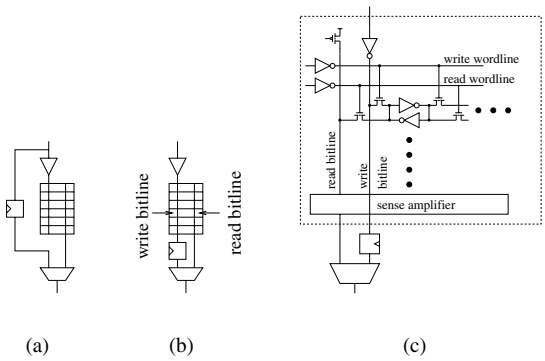


Figure 5. (a) Bypassing without overlapping. (b) Bypassing with overlapping. (c) Schematic of a write-through input buffer.

Impact on area. A cut-through crossbar clearly takes up less area than a matrix crossbar of the same scale.

3.3 Write-through input buffer

As pointed out in our analysis, buffer energy E_{buf} constitutes a significant portion of total router energy E_R, and is thus another target for power optimization.

In a typical wormhole router [8], when a flit enters an input port, it is written to the input buffer queue, and makes a request for switch traversal when it reaches the head of the queue. When the request is granted, the flit is read out of the input buffer, traverses the crossbar switch fabric and makes its way through the link to the next hop.

Bypassing input buffer is a common optimization for performance – when the flit arrives, if the buffer is empty, the flit heads straight to switch arbitration, and if it succeeds, the flit gets sent directly to the crossbar switch, circumventing the input buffers. Figure 5(a) illustrates how bypassing is typically implemented – with a separate bypass path connecting the router input port with the crossbar input port, and a flit-width register latching the data between consecutive pipeline stages of flit arrival and crossbar traversal.

The incoming flit is still written into the buffer so that if bypassing cannot be done, it will not be overwritten by the next flit. So bypassing only saves buffer read operations, at the expense of a separate bypass path. Since buffer write operations are not saved and the bypass path consumes extra energy and area, we propose an improved microarchitecture: write-through buffer, which realizes bypassing by overlapping the bypass path and buffer write bitlines, as sketched in Figure 5(b). Figure 5(c) shows the detailed schematic of a write-through buffer. Here the SRAM write bitlines are used as the bypass path and extend beyond the buffer. A write-through buffer essentially removes buffer read energy when bypassing can be done, at the expense of minimal hardware cost: pipeline registers and muxes.

Power analysis. Equation (1) can be augmented to reflect the power saving ability of write-through buffers:

$$E_{flit} = (E_{wrt} + \alpha \cdot E_{rd} + E_{arb} + E_{xb} + E_{lnk}) \cdot H \qquad (8)$$

where α is the probability of a flit being unable to bypass the input buffer.

By using queuing theory, we can derive an upper bound of the power savings of write-through buffers. The probability of the buffer being empty is:

$$p_0 = 1 - \frac{\lambda}{\mu}$$

where λ is the flit arrival rate and μ is the flit service or departure rate. Assuming no switch competition, so that $\mu = 1$, and we have:

$$\alpha = 1 - p_0 = \lambda$$

From [24],

$$\begin{aligned} E_{rd} &= E_{wl} + F(E_{br} + 2E_{chg}) \\ E_{wrt} &= E_{wl} + \delta_{bw} E_{bw} + \delta_{bc} E_{bc} \end{aligned}$$

where E_{wl} is wordline energy, F is flit size, E_{br} is read bitline energy, E_{chg} is pre-charging energy, E_{bw} is write bitline energy, E_{bc} is memory cell energy, and δ_{bw} and δ_{bc} are write bitline switching activity factor and memory cell switching activity factor respectively. Let r denote $\frac{E_{wrt}}{E_{rd}}$, so the energy consumed by a normal buffer is:

$$E_{buf} = E_{wrt} + E_{rd} = (r+1)E_{rd}$$

When using write-through buffers,

$$E'_{buf} = E_{wrt} + \lambda \cdot E_{rd} = (r+\lambda)E_{rd}$$

thus, the relative power saving is $\frac{1-\lambda}{r+1}$. When buffer size is large enough, $E_{br} \approx E_{bw}$ and $E_{wl}, E_{chg}, E_{bc} \ll E_{br}$. Assuming uniform random traffic, so that $\delta_{bw} = \frac{1}{2}F$ and $r = \frac{1}{2}$. At very low flit arrival rate ($\lambda \to 0$), the buffer power saving can reach 60%.

In reality, because on-chip networks tend to have small buffer size and also due to switch competition, write-through buffers achieve much less power savings than the upper bound.

Impact on performance. A write-through buffer can bypass buffer operations when the required conditions are met, otherwise it acts like a normal buffer, so it has no negative impact on network performance. In fact, a write-through buffer saves on buffer read/write delay at times, and improves network performance.

Impact on area. With overlapped bypass path and write bitlines, a write-through buffer only incurs marginal area overhead due to the additional registers and muxes.

3.4 Express cube

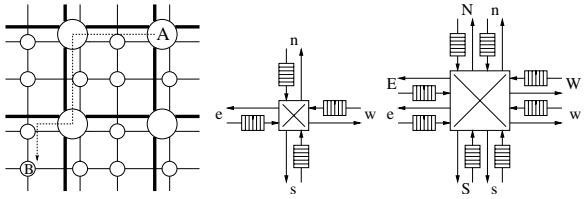
Express cube was first proposed in [7] to lower network latency by reducing average hop count. The main idea is to add extra channels between non-adjacent nodes, so that packets spanning long source-destination distances can shorten their network delay by traveling mainly along these express channels, thus reducing the average hop count.

We recognize that besides its performance benefit, an express cube can also reduce network power, since it reduces H, effectively removing intermediate router energy E_R completely. By targeting H, and eradicating E_R instead of lowering it as in the previous microarchitectural techniques, express cubes stand to reap the largest power savings.

Figure 6(a) shows a 2-D torus augmented with express channels. Express nodes are those connected by both local and express channels. Non-express nodes are referred to as local nodes. Express interval is the distance in hops between two adjacent express nodes. Figure 6(b) compares the size of a local node and an express node. An express node has twice as many network ports as a local node, which implies double buffer space and a larger crossbar, so an express node consumes more power than a local node. We take this into account in our power modeling of express cubes.

Power-performance analysis. Since the utilization of express channels heavily depends on network traffic, cycle-accurate simulation is needed to obtain actual power savings. Here, we analyze the reduction of hop count as an approximation of potential energy and delay savings, so we can explore the trade-offs in the design space of express cubes.

To simplify the analysis, we target ring topology. A normal ring with N nodes has average hop count $H = \frac{N^2}{4(N-1)}$ if N is even, or $H = \frac{N+1}{4}$ if N is odd. For a ring augmented



(a) A torus with express channels (shown bold), express interval is 2.

(b) Comparison of a local node and an express node. Lower case letters represent local ports, upper case letters represent express ports.

Figure 6. Express cube topology and microarchitecture.

with express channels, we define the following routing algorithm: given source node S and destination node D,

1. For S, locate its two closest express nodes: S_L and S_R. If S itself is an express node, $S_L = S_R = S$.
2. For D, locate D_L and D_R in the same fashion.
3. Compute the hop counts of five routes: $S \to S_L \to D_L \to D$, $S \to S_L \to D_R \to D$, $S \to S_R \to D_L \to D$, $S \to S_R \to D_R \to D$ and $S \to D$ directly. For each step of the route, if both nodes are express nodes, the route takes only express channels, otherwise it takes only local channels.
4. The route with the minimum hop count is a shortest path.

This routing algorithm is not very efficient, but it guarantees finding a shortest path.

By implementing this algorithm in C, we can compute the average hop count of an express ring and the reduction of the average hop count compared to a normal ring. Figure 7 shows some results. From the figure, we can see that express cubes have a wide range of hop count reduction (10-60%). For a fixed express interval, large networks tend to gain more benefit from express cubes since express channels have a higher chance of being used. For a fixed network size, the optimum express interval value lies between the smallest and largest valid interval values. This is because a large express interval can bypass more local nodes and yield more hop count reduction, but a large express interval also reduces the density of express nodes so that less traffic can benefit from them.

N-dimensional express cube routing is not as simple as the combination of N orthogonal express ring routings. For a 2-D express cube, the routing algorithm needs to locate the four closest express nodes to the source/destination node

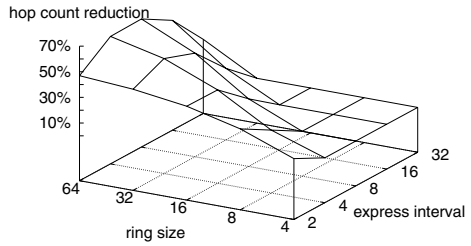


Figure 7. Relative hop count reduction of express cubes

and then enumerate 17 routes to find the shortest path. This also means more opportunities of hop count reduction. The dashed line in Figure 6(a) shows such an example, routing from node *A* to node *B*. This shortest path cannot be achieved by routing within one dimension, then another, but it does not lead to deadlocks as express channels and local channels belong to different virtual channels. Essentially, with the same express interval and same number of nodes per dimension, higher-dimensional express cubes can lead to larger hop count reduction.

Impact on area. Both express channels and larger express nodes require more area. However, their area overhead can be partially compensated by reducing flit size, and this is elaborated in the next section.

4 Simulation results

4.1 Experiment setup

We evaluated the four power-efficient network microarchitectures/architectures using our extension of Orion's power models plugged into PoPNet, a publicly available C++ network simulator [16]. In our discussion below, packet latency is defined as the number of cycles since the first flit of the packet is injected into the network until the last flit of the packet is ejected from the network. 0-load latency is the packet latency when the network is not congested at all, i.e. packet injection rate is close to 0. Throughput has several different definitions in different contexts, we use the definition that throughput is the data rate where packet latency reaches double the 0-load latency. While we generally use energy as a metric in our analysis, in this section, we use average total power since it is more relevant for a whole system.

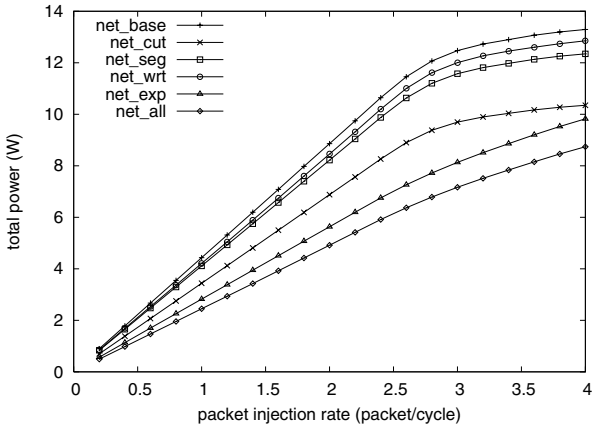
Using Raw [20], TRIPS [14] and the on-chip network proposed in [9] as references, we define the baseline network as follows: 2-D torus topology, 128-bit flits, 5 flits per packet, 2 virtual channels per port and 16-flit input buffer per virtual channel, and dimension-ordered routing. We assume that each virtual channel has a separate buffer, rather than two virtual channels sharing one buffer, to lower power consumption. We assume 1.2V voltage and 2GHz clock frequency at 0.1μm technology. The link length between adjacent nodes is 3mm. Based on the voltages suggested by [9, 25], we assume 300mV low-swing signaling for on-chip links. This baseline configuration is named **net_base**, and configurations extended with power saving techniques are as follows:

- **net_seg:** extends **net_base** with 2-segment segmented crossbars.
- **net_cut:** extends **net_base** with cut-through crossbars.
- **net_wrt:** extends **net_base** with write-through buffers.
- **net_exp:** extends **net_base** with express channels (express interval = 2). We assume that router pipeline is not lengthened for express nodes. For a fair comparison, we keep network bisection bandwidth constant. Since every other channel is augmented with express channels, a flit size of $128 \times \frac{2}{3} = 85$ bits is used for express cubes to equalize bisection bandwidth with that of a 128-bit wide torus. Packet size thus needs to be adjusted to $\frac{128\times5}{85} = 7.5$ flits so that each packet still contains the same number of bits. This is emulated by having 50% 7-flit packets, and 50% 8-flit packets.
- **net_all:** extends **net_base** with all four mechanisms, using express channels (express interval = 2), write-through input buffers, 2-segment segmented crossbars in express nodes, and cut-through crossbars in local nodes. Flit size and packet size are adjusted as in **net_exp**.

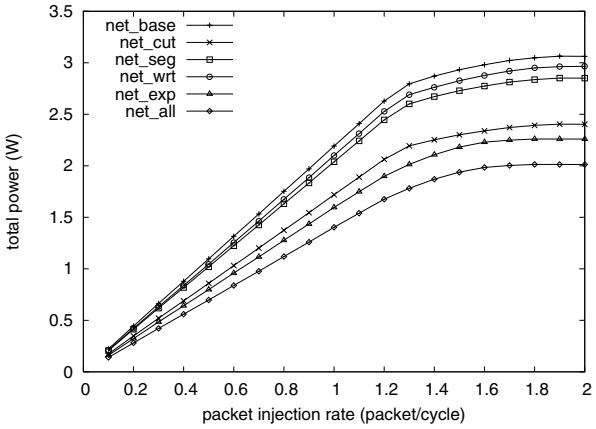
By keeping bisection bandwidth constant, **net_exp** uses the same link area as **net_base**. According to the area model built in our power model, an express node as configured in **net_exp** occupies 30% more area than a normal node as configured in **net_base**. With express interval being 2, 25% of network nodes are express nodes, so **net_exp** uses 7.5% more router area than **net_base**, and the overall network area overhead is even less. Since the other three mechanisms have no or negligible negative area impact, **net_all** has roughly identical area overhead as **net_exp**.

4.2 Synthetic uniform random traffic

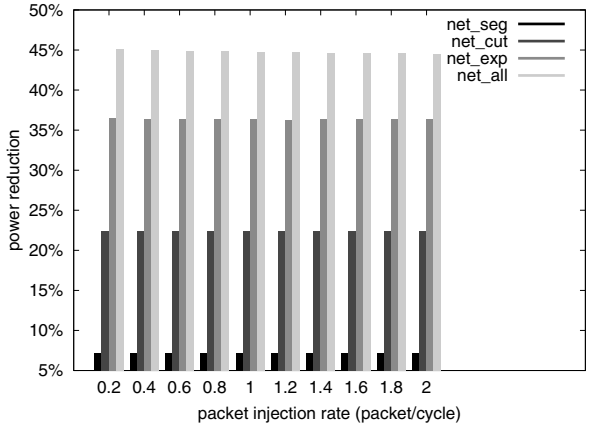
With uniform random traffic, each node injects packets to any other node with same probability and same data rate. Figures 8(a) and 8(b) show average total network power of all configurations for an 8×8 and a 4×4 torus network



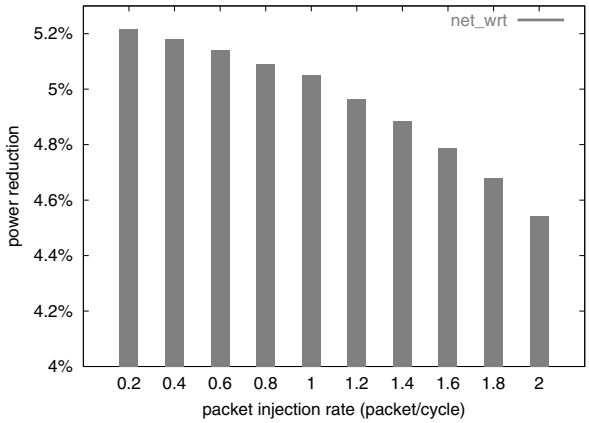
(a) 8×8 network power under random traffic.



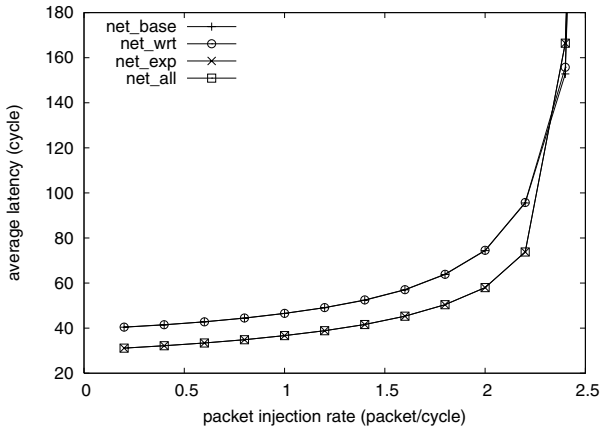
(b) 4×4 network power under random traffic.



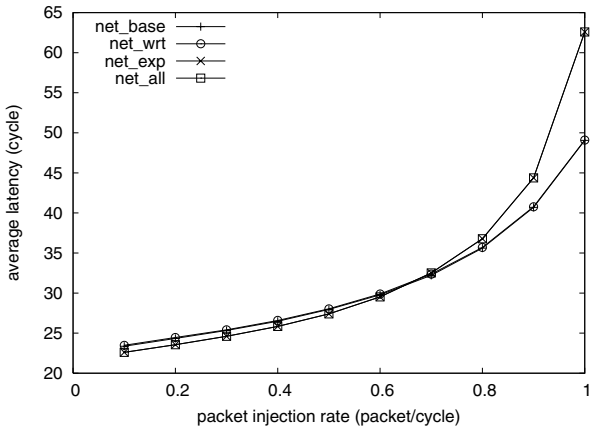
(c) 8×8 network power savings of 4 configurations.



(d) 8×8 network power savings of **net_wrt**.



(e) 8×8 network latency under random traffic.



(f) 4×4 network latency under random traffic.

Figure 8. Network power/performance under uniform random traffic.

respectively. Figure 8(c) shows power savings relative to **net_base** of **net_seg**, **net_cut**, **net_exp** and **net_all** for the 8×8 torus network. Since **net_wrt** has relatively low power savings, it is separately shown in Figure 8(d) with a small scale for better legibility. Figures 8(e) and 8(f) show average packet latency of all configurations for an 8×8 and a 4×4 torus network respectively.

Power savings. From Figure 8(c), we see that both segmented crossbars and cut-through crossbars result in noticeable power savings (7% and 22% respectively), and their power savings stay relatively invariant as network size changes, as shown in Table 3. Cut-through crossbars have higher power savings than segmented crossbars, which matches our analysis. However, cut-through crossbars do not scale well beyond 4×4, so the two microarchitectures can be used complementarily.

Write-through buffers only yield marginal power savings under uniform random traffic for the following two reasons:

- We observe that in our baseline configuration, input buffers consume 23% of total network power, less than crossbars (33%) and links (44%), so reduction in input buffer power does not translate to much total power reduction.

- When we derive the power saving upper bound in section 3.3, we assume large enough buffer size and no switch competition. Our baseline configuration has moderate buffer size, and switch competitions reduce chances of bypassing, both leading to less power savings.

Figure 8(d) shows the decreasing trend of write-through buffer power savings as packet rate increases, which implies more switch competitions.

Express cube is the winner. It single-handedly reduces network power by 27% for a 4×4 torus and 36% for an 8×8 torus. From Figure 7, we see that the average hop count reduction of express cubes is greater at larger network size, so the 8×8 torus receives higher power savings.

In Figure 8(a), express cubes have less power savings at high packet injection rates. This is because a network with express channels has larger saturation throughput, so the network can still sustain more traffic, which leads to more power consumption, while the baseline configuration is already saturated and its power consumption does not quite follow data rate increase. This phenomena is not obvious in Figure 8(b) because express channels are less effective for smaller networks.

Applying the other three techniques to an express cube network does not give as much power saving as they do to the baseline network. Cut-through crossbar, segmented crossbar and write-through buffer together produce an additional 8-9% power reduction beyond what is achieved with just express cubes. This is because the following two reasons:

- Our analysis in section 3.4 shows that some techniques produce higher power savings with larger flit sizes, and both **net_exp** and **net_all** have smaller flit size than other configurations.

- Express cubes reduce H, the number of routers traversed by a packet. Since all other three mechanisms target E_R, a lower H implies fewer opportunities to take advantage of them.

Impact on performance. According to our analysis, segmented crossbars have no performance impact. Cut-through crossbars have no performance impact either provided dimension-ordered routing is used. Express cubes reduce average latency by reducing average hop count, and write-through buffers can potentially improve performance by removing some buffer read operations.

From Figures 8(e) and 8(f), we see that express cubes reduce 0-load latency by 23% for an 8×8 torus and by 3.3% for a 4×4 torus. These simulation results match our analysis in section 3.4 that express cubes have better performance improvements for larger networks.

Write-through buffers have no noticeable performance improvement, which leads us to believe that the bypassing condition is rarely satisfied under uniform random traffic.

4.3 Chip multiprocessor traffic traces from the TRIPS benchmark suite

We ran a set of network traces extracted from a suite of sixteen benchmarks executed on one tile of the TRIPS CMPs to see the impact of the four power-efficient network microarchitectures/architectures on real traffic. Our simulator models the fairly unique topology of TRIPS [14].

Power savings relative to **net_base** are shown in Figure 9. The power savings of segmented crossbars and cut-through crossbars are almost invariant across all traces because they mainly depend on network configurations and are less traffic dependent. On the other hand, the power savings of express cubes vary greatly from trace to trace because the utilization of express channels is largely correlated with traffic patterns, which are determined by the applications. The power savings of write-through buffers vary slightly across traces, suggesting similar buffer occupancies across all traces. We are now conducting a more detailed investigation on the relations between the applications and the achievable power savings.

Table 3 summarizes the power savings of all experiments. For uniform random traffic, we use the average values of all data points before network congestion.

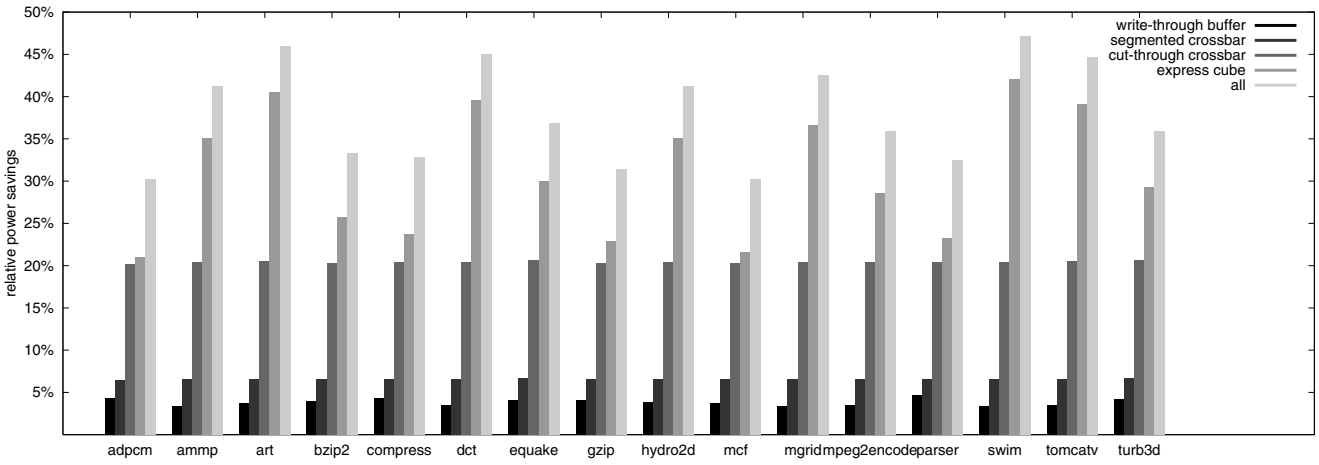


Figure 9. Power savings for network traces of the TRIPS benchmark suite.

Table 3. Average total network power savings (relative to net_base configuration).

	8×8 torus (random)	4×4 torus (random)	TRIPS traces
net_cut	22.4%	21.6%	20.4%
net_seg	7.2%	6.9%	6.6%
net_wrt	4.9%	4.5%	3.8%
net_exp	36.3%	27.2%	30.9%
net_all	44.9%	36.3%	37.9%

5 Related work

Power modeling. Patel *et al*. first noted the need to consider power constraints in interconnection network design, and proposed a power model of routers and links [12]. Wang *et al*. developed architectural power models for interconnection network routers and built Orion [24], a power-performance interconnection network simulator, which is used in this work as our power modeling framework.

Low power component designs. Most prior low power designs focus on reducing link power because that dominates power in off-chip interconnection networks. In 2000, Zhang, George and Rabaey proposed low-swing on-chip links [25], and Lee, Dally and Chiang proposed novel low power link designs [11]. Many low power bus encoding techniques have also been proposed and can be readily used on network links. However, these encoding techniques may not be readily applied to router components such as buffers and crossbars whose wire lengths are short relative to links, and thus unable to offset the overhead of encoding/decoding circuitry. While most previously proposed low power designs are circuit level techniques, our work targets the problem at the network microarchitecture level. To the best of our knowledge, this direction has not been previously explored for interconnection networks. Clearly, circuits and architectural techniques are synergistic and can lead to larger combined power savings.

Power management. Another dimension to lower power is to embed dynamic power management ability into routing and flow control policies. Shang, Peh and Jha explored dynamic voltage scaling with links to reduce network power dynamically [17]. They also developed PowerHerd [18], a framework which maintains global power budgets dynamically, sharing power budgets between adjacent nodes so as to maximize network performance while not exceeding peak power constraints. Chen and Peh targeted leakage power instead, with power-aware buffers that dynamically shut off in response to changes in utilization [5].

6 Conclusions

Unlike prior approaches that are primarily performance-driven, in this paper we adopt a power-driven perspective towards the design of on-chip network microarchitectures. As systems interconnected with on-chip networks become increasingly power-constrained, it is critical that we explore power-efficient network microarchitectures.

We first characterize the power profile of the on-chip network designs of two CMPs – the MIT Raw [20] and the UT Austin TRIPS [14], demonstrating that on-chip networks take up a significant percentage of total system power (36% in Raw). This motivated us to propose three power-efficient router microarchitectures and investigate the power efficiency of an existing network architecture, evaluating their power-performance-area impact with detailed power modeling and probabilistic analysis. We then evaluated the proposed network microarchitectures with synthetic as

well as real CMP benchmark traffic traces, realizing 44.9% power savings with uniform random traffic, and 37.9% with TRIPS CMP traces as compared to a baseline network microarchitecture based on current on-chip network designs. This substantial power saving is obtained with no degradation in network performance, and even improved performance in some cases.

Our study highlights the importance of a power-driven approach to on-chip network design. We will continue to investigate the interactions between traffic patterns and on-chip network architectures, and seek to reach a systematic design methodology for on-chip networks.

Acknowledgments

The authors would like to thank the MIT Raw group and the UT Austin TRIPS group for providing information on their on-chip networks and detailed benchmark traces. The authors would also like to thank all the anonymous reviewers for their invaluable comments. This work is partially funded by the DARPA MARCO Gigascale Silicon Research Center, NSF ITR grant CCR-0086031 and NSF CAREER grant CCR-0237540.

References

[1] F. Allen *et al.* Blue Gene: A vision for protein science using a petaflop supercomputer. *IBM Systems Journal*, 40(2):310–327, 2001.

[2] L. A. Barroso, K. Gharachorloo, R. McNamara, A. Nowatzyk, S. Qadeer, B. Sano, S. Smith, R. Stets, and B. Verghese. Piranha: A scalable architecture based on single-chip multiprocessing. In *Proc. International Symposium on Computer Architecture*, pages 282–293, 2000.

[3] L. Benini and G. D. Micheli. Powering networks on chips. In *Proc. International System Synthesis Symposium*, pages 33–38, 2001.

[4] J. Y. Chen, W. B. Jone, J. S. Wang, H.-I. Lu, and T. F. Chen. Segmented bus design for low-power systems. *IEEE Transactions on VLSI Systems*, 7(1):25–29, 1999.

[5] X. Chen and L.-S. Peh. Leakage power modeling and optimization in interconnection networks. In *Proc. International Symposium on Low Power Electronics and Design*, pages 90–95, 2003.

[6] W. J. Dally. Performance analysis of k-ary n-cube interconnection networks. *IEEE Transactions on Computers*, 39(6):775–785, 1990.

[7] W. J. Dally. Express cubes: Improving the performance of k-ary n-cube interconnection networks. *IEEE Transactions on Computers*, 40(9):1016–1023, 1991.

[8] W. J. Dally and C. L. Seitz. The torus routing chip. *Journal of Parallel and Distributed Computing*, 1(3):187–196, 1986.

[9] W. J. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. In *Proc. Design Automation Conference*, pages 684–689, 2001.

[10] J. S. Kim, M. B. Taylor, J. Miller, and D. Wentzlaff. Energy characterization of a tiled architecture processor with on-chip networks. In *Proc. International Symposium on Low Power Electronics and Design*, pages 424–427, 2003.

[11] M.-J. E. Lee, W. J. Dally, and P. Chiang. Low-power area-efficient high-speed I/O circuit techniques. *IEEE Journal of Solid-State Circuits*, 35(11):1591–1599, 2000.

[12] C. S. Patel, S. M. Chai, S. Yalamanchili, and D. E. Schimmel. Power constrained design of multiprocessor interconnection networks. In *Proc. International Conference on Computer Design*, pages 408–416, 1997.

[13] L.-S. Peh and W. J. Dally. Flit-reservation flow control. In *Proc. International Symposium on High Performance Computer Architecture*, pages 73–84, 2000.

[14] K. Sankaralingam, R. Nagarajan, H. Liu, C. Kim, J. Huh, D. Burger, S. W. Keckler, and C. R. Moore. Exploiting ILP, TLP, and DLP with the polymorphous TRIPS architecture. In *Proc. International Symposium on Computer Architecture*, pages 422–433, 2003.

[15] M. Sgroi, M. Sheets, A. Mihal, K. Keutzer, S. Malik, J. Rabaey, and A. Sangiovanni-Vincentelli. Addressing the system-on-a-chip interconnect woes through communication-based design. In *Proc. Design Automation Conference*, pages 667–672, 2001.

[16] L. Shang. PoPNet. http://www.ee.princeton.edu/~lshang/popnet.html, 2003.

[17] L. Shang, L.-S. Peh, and N. K. Jha. Dynamic voltage scaling with links for power optimization of interconnection networks. In *Proc. International Symposium on High Performance Computer Architecture*, pages 91–102, 2003.

[18] L. Shang, L.-S. Peh, and N. K. Jha. Powerherd: Dynamically satisfying peak power constraints in interconnection networks. In *Proc. 17th International Conference on Supercomputing*, pages 98–108, 2003.

[19] M. R. Stan and W. P. Burleson. Bus-invert coding for low power I/O. *IEEE Transactions on VLSI Systems*, 3(1):49–58, 1995.

[20] M. B. Taylor *et al.* The Raw microprocessor: A computational fabric for software circuits and general-purpose programs. *IEEE Micro*, 22(2):25–35, 2002.

[21] M. B. Taylor, W. Lee, S. Amarasinghe, and A. Agarwal. Scalar operand networks: On-chip interconnect for ILP in partitioned architectures. In *Proc. International Symposium on High Performance Computer Architecture*, pages 341–353, 2003.

[22] J. M. Tendler, J. S. Dodson, J. S. Fields, H. Le, and B. Sinharoy. Power4 system microarchitecture. *IBM Journal of Research and Development*, 46(1):5–25, 2002.

[23] H. Wang, L.-S. Peh, and S. Malik. Power characterization of the Raw and TRIPS on-chip networks. Technical report, Department of Electrical Engineering, Princeton University, 2003.

[24] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik. Orion: A power-performance simulator for interconnection networks. In *Proc. International Symposium on Microarchitecture*, pages 294–305, 2002.

[25] H. Zhang, V. George, and J. M. Rabaey. Low-swing on-chip signaling techniques: Effectiveness and robustness. *IEEE Transactions on VLSI Systems*, 8(3):264–272, 2000.