Segment Gating for Static Energy Reduction in Networks-On-Chip

Kyle C. Hale, Boris Grot, and Stephen W. Keckler
Department of Computer Sciences
The University of Texas at Austin
{khale, bgrot, skeckler}@cs.utexas.edu

ABSTRACT

Chip multiprocessors (CMPs) have emerged as a primary vehicle for overcoming the limitations of uniprocessor scaling, with power constraints now representing a key factor of CMP design. Recent studies have shown that the on-chip interconnection network (NOC) can consume as much as 36% of overall chip power. To date, researchers have employed several techniques to reduce power consumption in the network, including the use of on/off links by means of power gating. However, many of these techniques target dynamic power, and those that consider static power focus exclusively on flit buffers. In this paper, we aim to reduce static power consumption through a comprehensive approach that targets buffers, switches, arbitration units, and links. We establish an optimal power-down scheme which we use as an upper bound to evaluate several static policies on synthetic traffic patterns. We also evaluate dynamic utilization-aware power-down policies using traces from the PARSEC benchmark suite. We show that both static and dynamic policies can greatly reduce static energy at low injection rates with only minimal increases in dynamic energy and latency.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network architecture and design

General Terms

Measurement, Design

Keywords

Interconnection networks, leakage power, power optimization, network-on-chip (NOC), segment gating

1. INTRODUCTION

As designers integrate an increasing number of processing elements on a single die, Networks-On-Chip (NOCs) continue to gain appeal as efficient interconnection substrates for CMPs. Since researchers initially aimed to develop techniques that would reduce latency and boost throughput, several studies have addressed the factors that affect network performance such as topology [9], aggressive router microarchitectures [11], flow-control mechanisms [14], and routing algorithms [5]. As studies have shown that up to 36% of total chip power can be dissipated by the network [10], power constraints have recently come into prominence as a key aspect of NOC design.

Two components constitute chip power dissipation: dynamic switching power and static, or leakage, power. Until recently, researchers have utilized power saving techniques, such as Dynamic Voltage Scaling (DVS) [16] and Clock Gating [22], aimed at reducing dynamic power dissipation. However, as device sizes continue to scale down, static power due to gate, junction, and subthreshold leakage current [21] threatens to become dominant.

Our first contribution is to extend static power savings to additional components in the network. Previous work has focused solely on buffers [3, 13], neglecting potential savings from links and the router data and control paths. We find that a holistic approach that includes links, portions of the crossbar, arbiters, and buffers yields an appreciable reduction in static power consumption over previous schemes. Our approach is motivated by the observation that bandwidth is abundant in on-chip communication interconnects, leading to underutilization of available network resources on many workloads. This presents an opportunity to turn off links with little or no impact on the dynamic energy and latency in the rest of the network. Along with the link, arbitration logic, switch capacity, and buffering at the output port of the upstream node and input port downstream may be powered down as well. For the rest of the paper, we refer to this combination of link, logic, and storage at both ends of the link as a *segment*.

Our second contribution comprises a limit study that provides an upper bound on attainable leakage power savings for a set workload. Using that upper bound, we evaluate the same workload for various injection rates using two policies that power down segments statically. The first policy gates segments at random. The second one takes into account network port utilization in order to choose the segments best suited to enter a power-saving state.

As applications often exhibit irregular communication patterns that are difficult to predict or exploit through a static scheme, we also consider dynamic gating policies. These policies turn off segments when they are idle and wake them up on demand. We evaluate two such policies applied to a workload taken from the PARSEC benchmark suite [1]. The first policy assumes an aggressive mechanism for powering down segments, while the second emphasizes conservative behavior. We next analyze the reduction in leakage power that these policies offer over a completely ungated policy in which power saving mechanisms are not applied.

2. RELATED WORK

Several studies have explored the sources of leakage power. Powell et al. proposed a circuit technique called Gated-V_{DD} in which a gating transistor lies between one of the power rails and the circuit block, allowing the block to disconnect from the power supply [15]. In [6], Hanson et al. evaluate the efficacy of several mechanisms for reducing static energy in caches, including Gated-V_{DD}. Gated-V_{DD}, also called power gating, can be applied at various levels of granularity, e.g. to execution units [7] or individual SRAM cells [3].

In the context of NOCs, Chen and Peh analyze flit buffers as major sources of leakage and explore several policies for power-aware buffer designs [3]. Matsutani et al. employ Slow-Silent Virtual Channels to target both leakage and dynamic power consumption in the network [12, 13]. They combine power gating of virtual channels with DVS links to achieve an overall 58.2% reduction in network power.

Designers initially considered powering down links only for off-chip networks during periods of low utilization. Shang et al. proposed DVS for off-chip inter-node links [16]. Based on a link's history of utilization in the network, its voltage and frequency adjust dynamically to minimize dynamic power dissipation. Soteriou et al. extend the concept of regulating link power consumption to NOCs by suggesting that *on-chip* links be completely turned off during periods of inactivity [17]. Other studies statically construct connectivity graphs that guarantee that the network remains connected, even with some of the links powered down [18, 19]. Although Soteriou et al. show a potential 37.5% reduction in link power consumption [17], this reduction focuses on dynamic power, as do the above studies. Wang et al. also target dynamic power by proposing several power-aware crossbar microarchitectures, including the segmented crossbar [20].

In this paper we build on these techniques with a focus on static power consumption. Our approach combines the concept of on/off links [17] with power-aware buffers [3, 12, 13] and extends power-awareness to switches and arbiters.

3. SEGMENT GATING

We assume virtual channel routers with a speculative 2-stage pipeline and 4 VC buffers per input port. The crossbar switches employ a matrix organization with nMOS pass transistors at the cross-points. We further assume the use of two-level virtual channel and switch allocators with matrix arbiters at each level. When a link powers down, several components at the routers on each end transition into a power-saving state. At the downstream node, these include input buffers, the input side of the crossbar (word-line drivers and pass transistors), and first-level VC and switch arbiters. At the upstream node's output port, second-level VC and switch allocators can be powered down, along with the crossbar output port (bit-line drivers and pass transistors) and the output latch in front of the channel driver. We follow the methodology used by Chen et al. for power gating flit buffers [3]. For links, we apply fine-grained power gating

Table 1: Per-component energy values.

Component	Static (nJ/cycle)	Dynamic (nJ/flit)
Flit buffers	1.5	7.23
Crossbar	0.491	10.3
Allocators	0.215	0.7
Link	0.556	8.1

Table 2: Network configuration.

Topology	64-node mesh
Channels	128 bits wide, 1 cycle/link
Synthetic Workloads	uniform random, transpose, bit-compliment
PARSEC traces	blackscholes, bodytrack, fluidanimate, vips, x264. *Sim-medium* datasets
Router details	2-stage speculative pipeline, 5 ports, 4 VCs/port, 5 flits/VC

to the repeaters and drivers. We assume a mechanism for the crossbar that gates the drivers and the pass-transistors associated with a given input/output port. A similar mechanism gates the portion of an arbiter corresponding to a single segment. Although a detailed analysis of the aforementioned circuit techniques is required, we assume ideal operation for our experiments.

The values in Table 1 reflect the energy savings per component that result from gating a segment. The values for switch energy correspond to the sum of the energy values for an input port at the downstream node and an output port at the upstream node. The energy values for the allocator represent the aggregate energies at input (first level) and output (second level) arbiters for both virtual channel and switch allocators, combined. Note that buffers are responsible for just 55% of the overall leakage energy, highlighting the importance of targeting other microarchitectural components for maximum leakage reduction.

4. METHODOLOGY

Workloads: In our experiments we use both synthetic and real-world workloads on the network configuration shown in Table 2. We evaluate our static segment gating policies with uniform random, transpose, and bit complement traffic patterns. A synthetic workload consists of 1,000 packets injected by each node in the network. To make energy analysis tractable, we ignore the effects of contention. In Section 6, we evaluate potential policies using communication traces taken from the PARSEC suite [1] with the same network configuration.

Tools: We developed an off-line analysis tool that we use for evaluating various static segment gating policies. This tool represents the mesh as a directed graph with two edges between each pair of nodes, where gating a segment means removing an edge from the graph. We derive hop counts by calculating shortest path lengths between source-destination pairs. Depending on the available edges, these shortest paths may be longer than minimal manhattan routes. The tool outputs mean, maximum, and minimum hop counts resulting from removing the specified number of edges for several experiments.

The random segment gating policy removes edges from the graph using a stochastic process while ensuring that the resulting graph remains connected. Each experiment can result in a different graph due to the random selection process and because there may exist multiple shortest paths between a pair of nodes. Each run of the algorithm represents a sample; results in Section 5 were collected using 50 random samples.

The intelligent segment gating scheme uses edge utilization as the basis for the selection process. Edge weights are computed based on the number of communicating nodes that utilize each link. At each step, the policy selects a segment with minimum utilization, breaking ties randomly. Affected flows are re-routed over the remaining links, edge weights are recomputed, and a new iteration begins. Due to the stochastic element in the selection process, each experiment consists of 30 samples.

We evaluate dynamic segment gating policies using real-world application traces from the PARSEC 1.0 suite [1], gathered via the M5 full-system simulator [2] and replayed in our custom network simulator that captures detailed microarchitectural events. We used a record of these events, which include link and port utilization intervals, to apply several policies off-line.

Routing: For static segment gating policies, we assume a topology-aware routing algorithm that always chooses the minimal path given a connectivity graph. The gating policies are evaluated using static off-line analysis which does not take issues of contention or deadlock into account. In practice, suspected deadlock situations can be detected using a per-VC watchdog timer and resolved via a deadlock resolution protocol.

For the dynamic policy space, we use dimension-order routing (DOR). Links are power-gated and woken-up on demand, and no misroutes take place.

Power Model: Table 1 shows per-component energy values, derived from ORION 2.0 [8] assuming a 45nm process, 0.8V V_{DD}, and a 2GHz clock frequency. By combining values from Table 1, workload characteristics and hop count data, we obtain measurements of energy savings. Dynamic energy is summed over all message traversals. The dynamic energy for each message traversal equals the number of hops for the message multiplied by the dynamic energy incurred at each hop. Static energy is calculated by multiplying the number of active ports in the network, the static energy per cycle per port, and the sum of cycles during which the port is powered on.

5. LIMIT STUDY EVALUATION

This section focuses on the benefits of segment gating through a limit study. We obtain an upper bound on the possible energy savings from turning off segments and compare it to our static policies. We do not consider the effects of contention, leaving that for future work.

5.1 Optimal Dynamic Scheme

In order to measure the effectiveness of static policies, we devise an ideal gating scheme in which no energy or latency overhead is incurred by powering down segments. This scheme shuts down a segment during any period of inactivity with no latency or energy overhead, therefore maximizing static energy savings. We assume a mechanism similar to that used by Soteriou et al. where links are reactivated *on-demand* [19], but here entire segments are signaled to wake-up as soon as the physical output channel is known. Under our assumptions (perfect gating of inactive segments and no contention), energy figures are independent of injection rates because segments only leak when they experience switching activity. Thus, static energy for the optimal dynamic scheme is proportional to dynamic energy since both, in this case, depend on the total number of segments traversed by a message.

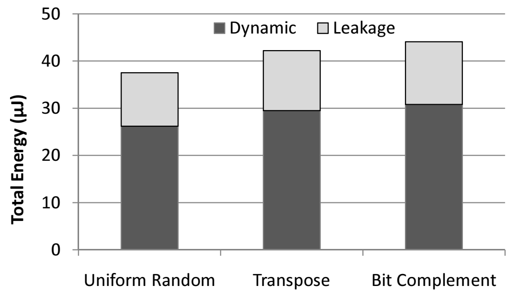


Figure 1: Optimal Energy Consumption

Figure 1 shows optimal energy consumption for synthetic traffic patterns. In general, dynamic power dominates because transistors only leak when dynamic events take place, e.g. when a message traverses a router or link. As expected, static energy represents nearly 30% of switching energy. In reality, we must pay both an energy and latency cost for gating segments, so we expect that these measures are unattainable in practice. However, because real systems leave a great deal of idle hardware on for long periods of time, we can extract significant savings in leakage by establishing practical policies that power-gate segments in the network. The next section illustrates two specific static policies for segment gating.

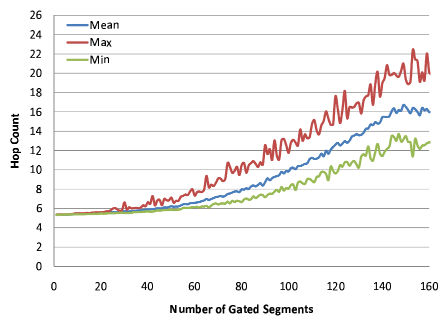
5.2 Static Scheme with Random Segment Selection

We start with a simple static policy that selects which segments to gate at random. We employ the analysis tool described in Section 4 to generate hop-counts produced by removing edges from the graph representing the network. Both of the following static policies assume that a gated segment remains gated for the duration of the workload.

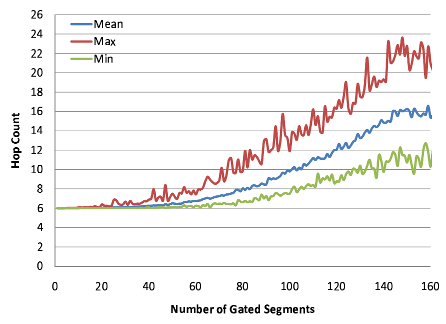
Figure 2 shows the mean, maximum, and minimum hop counts for different traffic patterns. When edges are removed at random from the graph, the hop count grows very slowly. For all three traffic patterns, 20 edges can be removed with negligible increase in the average hop-count – and therefore little latency and dynamic energy overhead. When half of the network links are powered down, the average hop count approximately doubles. Finally, of the 224 total segments (links) in a 64-node mesh, at most 161 can be disabled while maintaining full connectivity at the cost of 2-4x increase in hop count.

Figure 3 shows the effect of random segment gating on network energy consumption with different number of gated links and injection rates. As our workload consists of a fixed number of packets, lower injection rates result in longer completion times. In turn, long execution times lead to large leakage energy expenditures, as components stay 'on' for extended periods of time.

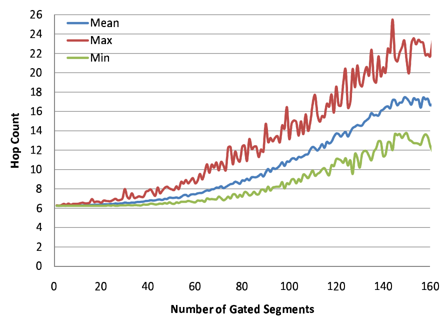
As we expect, leakage energy drops almost linearly with



(a) Uniform Random Traffic

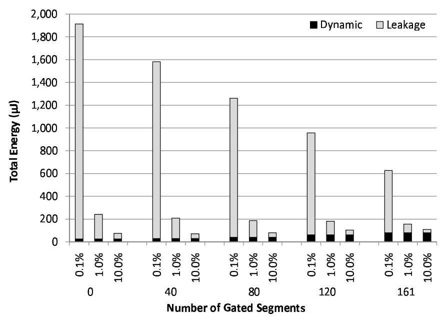


(b) Transpose Traffic

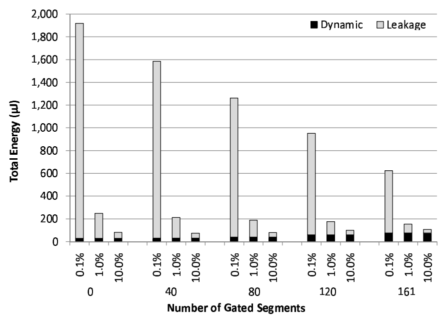


(c) Bit Complement Traffic

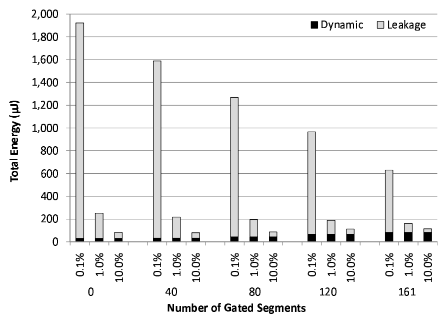
Figure 2: Hop Counts for Synthetic Traffic Patterns



(a) Uniform Random Traffic



(b) Transpose Traffic



(c) Bit Complement Traffic

Figure 3: Total Energy for Synthetic Traffic Patterns at Different Injection Rates

the number of gated segments. However, disabling links causes path lengths to increase, in turn causing a corresponding rise in dynamic energy consumption. Contention caused by a large number of disabled links would be expected to prolong workload execution, especially at higher injection rates. This will have a detrimental effect on leakage power consumption which is not captured with our existing methodology.

Although in the best case we see energy consumption nearly twice that of the optimal policy, random segment selection neglects the underlying characteristics of the traffic patterns.

5.3 Static Scheme with Utilization-Based Segment Selection

A number of applications have predictable communication characteristics, which we can leverage to maximize static energy savings by turning off infrequently used links. For workloads with a high degree of communication regularity, a static policy that takes link utilization into account might offer a generous reduction in static energy consumption. This policy, of course, requires prior knowledge of the workload's communication patterns.

We identify two techniques to power down segments based on utilization. The first simply powers down all segments with a utilization of zero. Traffic patterns like *bit-complement* and *transpose* lend themselves well to this strategy, as they idle 92 and 100 network ports, respectively, out of 224 total ports. This allows us to turn off close to half of the network

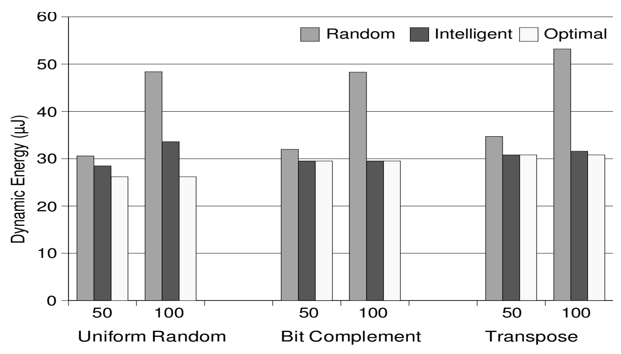


Figure 4: Dynamic Energy Incurred from Utilization-aware Segment Gating

with no increase in dynamic energy or latency. The second technique affords additional leakage energy savings by turning off some of the remaining links. By preferentially gating the least-utilized segment in each iteration, the policy tries to minimize the number of flows that have to be rerouted.

Figure 4 compares dynamic energy dissipated by random and utilization-aware static policies with 50 and 100 gated segments, respectively. Because the workload consists of a fixed number of messages, dynamic energy is independent of injection rate and only increases due to misroutes attributed to gated segments. Static energy is not shown as it remains the same for a given number of gated segments.

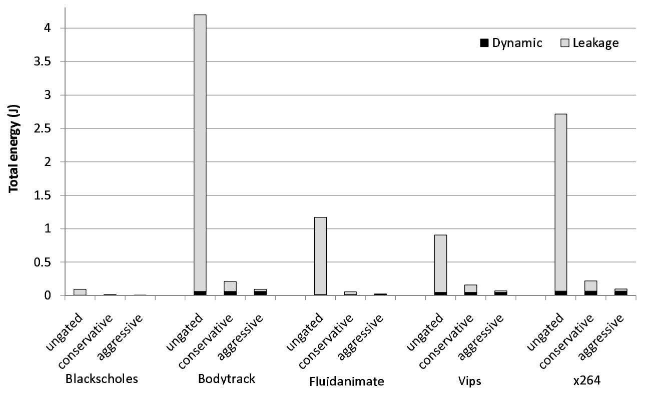


Figure 5: Energy Consumption for PARSEC benchmark traces using dynamic policies

Compared to a random gating policy, utilization-based gating yields considerable reductions in dynamic energy consumption by decreasing the incidence of misroutes. With 100 gated segments, dynamic energy is reduced by 31%, on average, over random gating. The energy savings are highest on the *bit-compliment* permutation, with nearly 40% of dynamic energy (22% of the total energy) saved.

Compared to an optimal policy that never routes a packet non-minimally, the utilization-based scheme incurs a maximum energy overhead on the *uniform random* pattern. On this traffic pattern, packets with non-minimal path lengths contribute 28% (9%) in dynamic energy with 100 (50) links gated. Random gating, however, leads to an 85% (17%) energy overhead due to misroutes. Thus, by intelligently selecting segments to power down based on workload characteristics, we can reduce the network's static energy consumption while minimizing the impact on dynamic energy.

6. DYNAMIC POLICY SPACE DISCUSSION

Although static policies offer a reduction in leakage energy, they cannot accommodate workloads that have dynamic communication patterns. In reality, many real applications have the characteristic of bursty traffic, where nodes may remain idle for thousands of cycles and then suddenly begin injecting packets into the network [4].

In order to dynamically power down segments in a practical setting we employ mechanisms similar to those used by Matsutani et al. [13]. During route computation, once a downstream input port is selected for a message, the output channel and the appropriate output line of the crossbar are activated and a wake-up line is asserted, signalling the downstream node to power up the VCs, arbiters, and crossbar input lines corresponding to that input port. The costs of the wake-up and power-down transitions in terms of latency and energy are discussed in detail by Hu et al. [7].

The decision to turn off a segment is based on the number of cycles that a given link is idle and represents an important policy decision. In addition, there is a period of time that segments must be gated in order to amortize the fixed energy costs of power-down and wake-up transitions, called the break-even point [7].

In our experiment we choose two different policies to compare against the baseline, which employs no power saving mechanisms. Policies are applied to workloads represented by traces generated from the PARSEC benchmark suite [1]. The first policy assumes an aggressive idle period, where only 2 cycles of inactivity on a link must pass before gating the segment. We also assume a small, 10-cycle *break-even* interval, the time in which a segment must remain gated in order to amortize the energy costs of powering on and off. The conservative policy waits longer to gate segments, with a 10-cycle idle period and a 50-cycle break-even interval. Figure 5 shows the total energy for the two policies and the baseline ungated scheme.

These results show that after applying the aggressive policy, segments spend 97-99% of total cycles in the power-saving state. For the conservative policy, 88-97% of cycles are spent gated. These gated cycles correspond directly to static energy savings. The conservative scheme is more practical in that it reflects a break-even point that more accurately represents the 2GHz routers we assume. However, under workloads like these traces with sparse communication, the aggressive policy maximizes static energy savings. We point out that longer-running workloads offer more opportunities for reduction in leakage energy.

Although our results show a huge potential for leakage energy reduction, we do not account for contention or performance. Due to wake-up and power-down latencies, these policies may hamper performance by causing router pipeline stalls. Further, different workloads might show affinity for different policies. In general, workloads showing high injection rates will perform poorly using an aggressive scheme, and very little energy savings result because segments turn on and off repeatedly, not allowing time to reach the break-even point. For example, grid solvers have nodes communicating with nearest-neighbor traffic very frequently. Suppose a 2-cycle idle period is chosen and nodes communicate with their neighbors every 4 cycles. This will result in very poor performance and zero energy savings. Thus, a more conservative policy would be more appropriate. Because of the disparities in real-world communication patterns, we point out that a scheme in which idle periods vary dynamically would be favorable, allowing for the network to maintain a high degree of flexibility relative to the workload.

7. CONCLUSIONS

With real applications showing a great deal of network inactivity, the potential for saving static energy is quite high. We have shown that with intelligent static schemes we can reduce leakage energy consumption considerably. By taking into account link utilization, we can minimize dynamic energy overhead due to segment gating. If we apply an aggressive dynamic policy to real application workloads showing sparse communication characteristics, there is potential to save leakage energy for up to 99% of total cycles.

The difficulty of maximizing savings in the network lies in the fundamental difference between computational hardware and infrastructural hardware. Computational elements like cores are easier to target with techniques like power-gating as their behavior is predictable. However, infrastructure like the interconnection network is much harder to predict because communication can vary greatly with different applications. Therefore, making coarse-grained decisions about which portions of the network are essential is a difficult problem. In general, our results provide motivation for *introspective* networks that employ self-regulation in order to eliminate wasted energy.

There remain many aspects of introspective network design that must be investigated. First, circuit techniques for efficiently gating elements on the router control paths must be explored in detail. Next, we aim to expand our current analysis to account for network contention and to include details concerning the performance impacts of segment gating. Further, we intend to perform an in-depth analysis that will clearly define the regimes in which segment gating is beneficial. Finally, since segment gating can potentially create an amorphous network, we plan to explore its applications to fault tolerant systems.

8. REFERENCES

[1] C. Bienia, S. Kumar, J. P. Singh, and K. Li. The PARSEC Benchmark Suite: Characterization and Architectural Implications. In *International Conference on Parallel Architectures and Compilation Techniques*, pages 72–81, October 2008.

[2] N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt. The M5 Simulator: Modeling Networked Systems. *IEEE Micro*, 26(4):52–60, 2006.

[3] X. Chen and L.-S. Peh. Leakage Power Modeling and Optimization in Interconnection Networks. In *International Symposium on Low Power Electronics and Design*, pages 90–95, August 2003.

[4] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[5] C. J. Glass and L. M. Ni. The Turn Model for Adaptive Routing. In *International Symposium on Computer Architecture*, pages 278–287, May 1992.

[6] H. Hanson, M. S. Hrishikesh, V. Agarwal, S. W. Keckler, and D. Burger. Static Energy Reduction Techniques for Microprocessor Caches. *IEEE Transactions on VLSI Systems*, 11(3):303–313, June 2003.

[7] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. Zyuban, H. Jacobson, and P. Bose. Microarchitectural Techniques for Power Gating of Execution Units. In *International Symposium on Low Power Electronics and Design*, pages 32–37, August 2004.

[8] A. Kahng, B. Li, L.-S. Peh, and K. Samadi. Orion 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration. In *Design, Automation and Test in Europe*, pages 423–428, April 2009.

[9] J. Kim, J. Balfour, and W. Dally. Flattened Butterfly Topology for On-chip Networks. In *International Symposium on Microarchitecture*, pages 172–182, December 2007.

[10] J. S. Kim, M. B. Taylor, J. Miller, and D. Wentzlaff. Energy Characterization of a Tiled Architecture Processor with On-Chip Networks. In *International Symposium on Low Power Electronics and Design*, pages 424–427, August 2003.

[11] A. Kumar, P. Kundu, A. P. Singh, L.-S. Peh, and N. K. Jha. A 4.6Tbits/s 3.6GHz Single-Cycle NoC Router with a Novel Switch Allocator in 65nm CMOS. In *International Conference on Computer Design*, pages 63–70, October 2007.

[12] H. Matsutani, M. Koibuchi, H. Amano, and D. Wang. Run-time Power Gating of On-Chip Routers Using Look-Ahead Routing. In *Asia and South Pacific Design Automation Conference*, pages 55–60, January 2008.

[13] H. Matsutani, M. Koibuchi, D. Wang, and H. Amano. Adding Slow-Silent Virtual Channels for Low-Power On-Chip Networks. In *International Symposium on Networks-on-Chip*, pages 23–32, April 2008.

[14] G. Michelogiannakis, J. D. Balfour, and W. J. Dally. Elastic-Buffer Flow Control for On-Chip Networks. In *International Symposium on High-Performance Computer Architecture*, pages 151–162, February 2009.

[15] M. Powell, S.-H. Yang, B. Falsafi, K. Roy, and T. N. Vijaykumar. Gated-Vdd: a Circuit Technique to Reduce Leakage in Deep-Submicron Cache Memories. In *International Symposium on Low Power Electronics and Design*, pages 90–95, July 2000.

[16] L. Shang, L.-S. Peh, and N. K. Jha. Dynamic Voltage Scaling with Links for Power Optimization of Interconnection Networks. In *International Symposium on High-Performance Computer Architecture*, pages 91–102, February 2003.

[17] V. Soteriou and L.-S. Peh. Dynamic Power Management for Power Optimization of Interconnection Networks Using On/Off Links. In *International Symposium on High Performance Interconnects*, pages 15–20. IEEE Computer Society, August 2003.

[18] V. Soteriou and L.-S. Peh. Design-Space Exploration of Power-Aware On/Off Interconnection Networks. In *International Conference on Computer Design*, pages 510–517, October 2004.

[19] V. Soteriou and L.-S. Peh. Exploring the Design Space of Self-Regulating Power-Aware On/Off Interconnection Networks. *IEEE Transactions on Parallel and Distributed Systems*, 18(3):393–408, March 2007.

[20] H. Wang, L.-S. Peh, and S. Malik. Power-driven Design of Router Microarchitectures in On-chip Networks. In *International Symposium on Microarchitecture*, pages 105–116, December 2003.

[21] N. Weste and D. Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*. Addison Wesley, 3rd edition, May 2004.

[22] Q. Wu, M. Pedram, and X. Wu. Clock-gating and its application to low power design of sequential circuits. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 47(3):415–420, March 2000.