Design-Space Exploration of Power-Aware On/Off Interconnection Networks

Vassos Soteriou and Li-Shiuan Peh
Department of Electrical Engineering
Princeton University, Princeton, NJ 08544
{soteriou,peh}@ee.princeton.edu

***Abstract*— With power a major limiting factor in the design of scalable interconnected systems, power-aware networks will become inherent components of single-chip and multi-chip systems. As communication links consume significant power regardless of utilization, we propose and investigate power-aware networks whose links are turned on and off in response to bursts and dips in traffic. We explore the design space of such on/off networks, outlining a 5-step design methodology along with solutions at each step that can form the building blocks of numerous designs. Two specific designs targeting links with substantially different on/off times are then presented and evaluated. Our simulations show that up to 54.4% power savings can be achieved along with at most 7.5% increase in latency.**

I. Introduction

Interconnection network fabrics have lately been used in a wide range of communication systems – multiprocessor servers [1], terabit Internet routers [2], clusters [3], server blades [4], and multiprocessor systems-on-a-chip [5] [6]. In pursuit for wider bandwidth and higher communication efficiency, traditional buses are being replaced by network fabrics as the principal interconnect that incorporate high speed router cores and communication links with data rates on the order of several Gb/s to match bandwidth requirements. With the tight power constraints faced by these systems, and interconnection networks dissipating a significant portion of total system power (36% in the MIT Raw CMP [7], 33% in the Avici Internet router [2], and 20% in the Alpha 21364 microprocessor [1]), it is now timely to explore power-aware networks.

In board-to-board and multi-chip networks, links are already consuming substantial power. They take up 64% of the power budget of the IBM 8-port 12X switch (each of the eight ports interfaces 12 2.5Gbps links that dissipate 2.5W each, with the entire switch consuming 31W on average) [4]. In addition, these links dissipate substantial power even when no data is being transmitted (average-case power of the IBM InfiniBand 12X link is almost identical to its worst-case power). In on-chip networks, several new link architectures are recently proposed for global wires, replacing full-swing repeatered wires [8] [9]. Due to features such as differential and pulsed current-mode signalling, these links have power profiles that are similarly invariant to utilization. As a result, slowing network traffic has the interesting contrary effect of actually increasing power since a link consumes the same power when idle. It is thus apparent that power-aware networks that turn links on/off in response to varying utilization can harvest considerable power savings.

This paper explores the design space for on/off networks and proposes a 5-step design methodology for such networks. At each step, we propose building blocks that can be assembled to form numerous power-aware on/off network designs. Two specific designs that address different tradeoffs of latency vs. power savings are then described and evaluated. With faster on/off links where it takes 1,000 cycles to turn off a link and another 1,000 cycles to turn it back on, we show that up to 54.4% link power reduction with SPEC2000 and MediaBench [10] applications running on a network-on-chip can be achieved. These are accompanied with minimal impact on performance, ranging from 0.3% to 7.5% increase in latency. Even with the large 10,000 cycle on/off delays of today's box-to-box and board-to-board links [11], our design enables up to a substantial 33.5% link power savings with no degradation in network throughput. Network latency is increased by 4-5 cycles on average, a delay that can be readily hidden by the network interface delay that spans hundreds of cycles [12], such as that in cluster interconnects like InfiniBand. To the best of our knowledge, this is the first work that explores the design space of power-aware on/off networks.

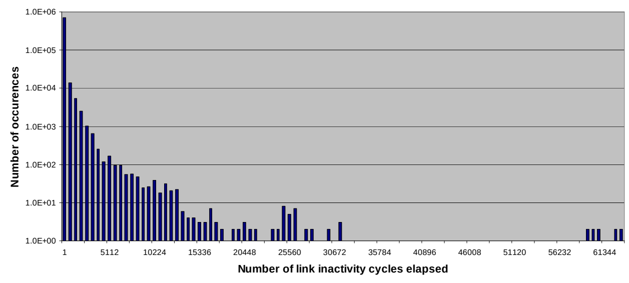

Fig. 1. Histogram of link inactivity periods of the *ammp* trace benchmark from the TRIPS CMP over $300k$ cycles of simulation time.

The structure of this paper is as follows. Section II motivates the potential of on/off networks, exploring diverse network traffic behavior assuming oracle knowledge. Following, Section III explores the design space of on/off networks and presents our 5-step design methodology. Section IV describes a number of designs while Section V presents the evaluation results. Finally, Section VI discusses prior related research and Section VII wraps up the paper.

II. Motivation

A. Inactivity Periods

Communication data traversing an interconnection network exhibits high variance both spatially and temporally, giving rise to inactivity or "dead" periods at various links in the network. An inactivity period starts from when the last flit[1] departs till the instant when a new flit starts using the same link.

Fig. 1 shows the histogram of the inactivity periods of the `ammp` traffic trace benchmark running on a 5×5 on-chip inter-ALU network of the Austin TRIPS CMP [6] over $300k$ cycles across all links. It affirms the presence of high variance in network traffic – we see inactivity periods from a cycle to as long as $65k$ cycles.

B. Potential Benefits

To motivate the potential of our approach, we simulate an "oracle" predictor that has *perfect* information about current and future network traffic. It precisely puts a link to sleep when it knows that it will not be used for a period longer than twice the on/off link delay. The predictor wakes up the link just-in-time enabling link power savings *without any performance impact*. We run simulations with our oracle predictor using real traffic traces from the SPEC2000 and MediaBench [10] benchmarks running on the TRIPS CMP and using synthetic self-similar traces on an 8×8 torus. Section V-A provides a description of the simulation environment. Fig. 2 shows the ideal fraction of time during which all links in the network can turn off without impacting performance for the TRIPS trace

[1] Flit stands for flow control unit, a fixed-size segment of a packet.

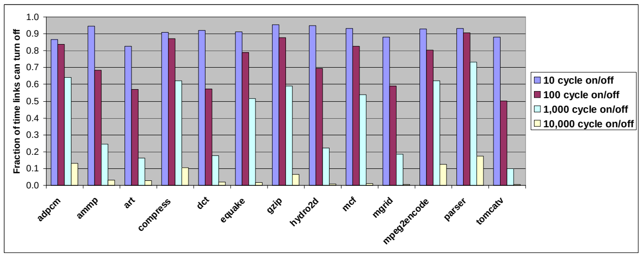

Fig. 2. "Oracle" approach: ideal fraction of time during which links can turn off *without* any performance degradation with varying on/off link transition delays for various TRIPS CMP benchmark traces with Opt-Y routing.

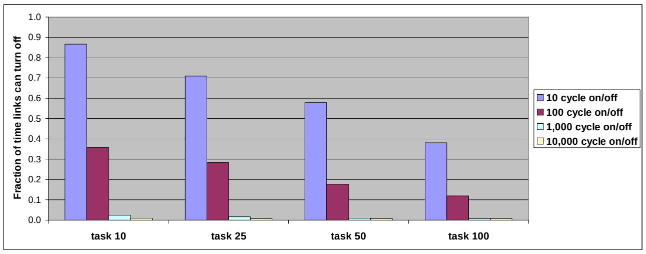

Fig. 3. "Oracle" approach: ideal fraction of time during which links can turn off *without* any performance degradation with varying on/off link transition delays for synthetic self-similar traffic for an 8 × 8 torus network with Opt-Y routing.

benchmarks. Note that as the link on/off transition delay increases, the opportunities to turn links off drop.

With shorter link on/off transition delays the opportunities to turn links off with our "oracle" predictor exceed the potential of 86% for all traffic trace benchmarks shown here. Note that the oracle predictor is *not necessarily an upper bound* on the achievable power savings. Rather, the percentage values shown here are subject to the constraint that we only turn a link off when we are subject to zero performance degradation. If we turn a link off while "violating" the twice the on/off link delay rule explained earlier, we may further reduce link power consumption at the expense of traffic delays.

Fig. 3 shows the ideal fraction of time during which links in the network can turn off without impacting performance for a synthetic self-similar traffic model with 10, 25, 50 and 100 concurrent tasks that is reflective of traffic in clusters (See Section V-B.2 for a description of the model). This traffic exhibits substantially less temporal variance than that of the TRIPS CMP traces, therefore the inactivity periods are sparser and shorter with narrower power savings opportunities. Also, as the number of concurrent tasks increases, the variance in network traffic is reduced, leading to lower potential link power savings.

III. Design-Space Exploration of Power-Aware On/Off Networks

In the design of an on/off network, a designer is faced with five key decisions in the face of tight power constraints as well as performance targets:

- Which links to turn off?
- How to route packets when links are off?
- When to put a link to sleep?
- When to wake up a link?
- How to implement on/off within a router pipeline?

Here, we propose a 5-step methodology that guides designers through these respective decisions. At each step, we will illustrate our methodology through the proposal of specific design components. Designers can then pick and choose components at each step to form an on/off network design. Fig. 4 summarizes the design space we uncover for on/off networks.

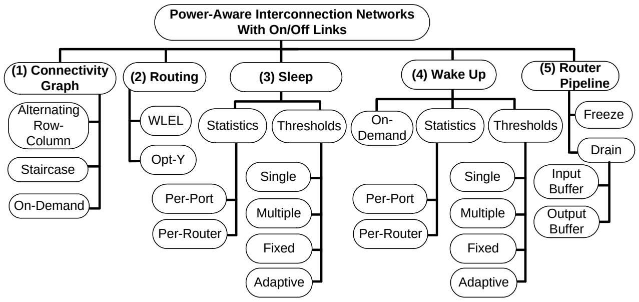

Fig. 4. Design space of power-aware on/off networks based on our 5-step methodology.

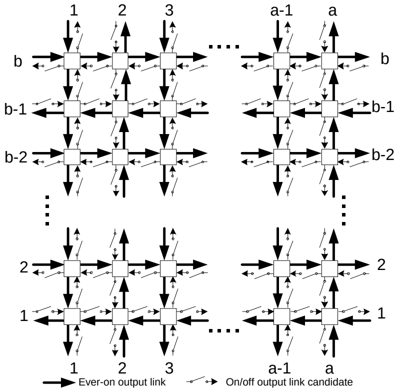

Fig. 5. $a \times b$ torus alternating row-column connectivity graph with restricted on/off link candidates shown as switches.

A. Which Links to Turn Off?

A network topology with all links on is a fully connected graph. We propose an on/off connectivity graph – one that classifies the fully connected graph into two sets – *ever-on* links (that must remain always on) and *on/off link candidates* (links that can be turned on/off). The on/off connectivity graph must guarantee full network connectivity through its ever-on links, i.e. each node can communicate with any other node in the network even when all on/off link candidates are shut down, so that no network partitions are created.

Fundamentally, ever-on links act as escape paths that can be used to reroute packets when on/off link candidates are shut down, and also conceal delays when traffic levels require the reactivation of an inactive on/off link candidate by providing alternative routes.

*1) **Alternating Row-Column Graph**:* This on/off connectivity graph constrains the two on/off link candidates to be on different dimensions at every router. This ensures alternating rows and columns of on/off link candidates and ever-on links in the x and y dimensions respectively. Fig. 5 depicts an alternating row-column connectivity graph for tori. For meshes, all links along the perimeter of the network are designated ever-on for deadlock freedom [13].

*2) **Staircase Graph**:* The staircase graph constrains the two on/off link candidates to be on the same dimension at every router, either both along x or along y. This gives rise to a staircase of on/off link candidates and ever-on links; an upwards staircase and an immediate downwards staircase structure across the entire network. Fig. 6 shows a torus staircase connectivity graph. The same graph applies to meshes, minus wraparound links along the borders.

*3) **On-demand Graph**:* With links that can be turned on/off rapidly, disconnectivity is no longer a crippling concern – since links can respond quickly to abrupt changes in traffic utilization, promptly waking up on demand. An on-demand connectivity graph where all links are on/off link candidates becomes feasible.

*4) **Discussion**:* Essentially, the on/off connectivity graph dictates the maximum potential power-performance of an on/off network. For instance, the alternating and staircase connectivity graphs limit

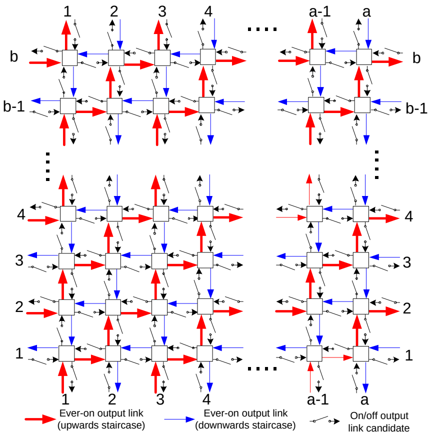
1
2
3
4
....
a-1
a
b
b-1
4
3
2
1
....
Ever-on output link (upwards staircase)
Ever-on output link (downwards staircase)
On/off output link candidate

Fig. 6. $a \times b$ torus staircase connectivity graph with restricted on/off link candidates shown as switches.

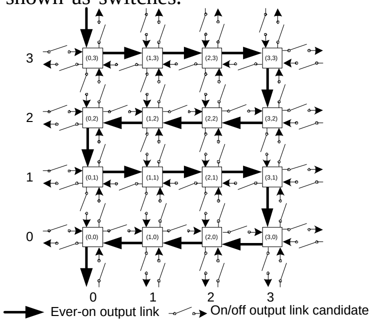

Fig. 7. 4×4 torus connectivity graph incorporating a minimum spanning tree with ever-on links depicted as bold solid lines.

the number of on/off link candidates per router to two. Hence, maximum potential network link power savings is 50%. Sparser on/off connectivity graphs with fewer ever-on links will lead to higher potential power savings; a connectivity graph can be built in the form of a minimum spanning tree as shown in Fig. 7. However such a graph, with a worst case hop count of $k^2 - 1$ (when all on/off link candidates are off), can degrade performance substantially. With an on-demand connectivity graph, a 100% network link power savings can potentially be attained since all links can be off.

TABLE I
WORST CASE HOP COUNTS IN A $k \times k$ NETWORK

	Mesh	Torus
Alternating Row-Column	$3 \times (k-1)$	$3 \times (\frac{k}{2})$
Staircase	$2k-1$	k
On-demand	inf	inf

Table I shows their worst-case hop counts. The proposed staircase and alternating row-column graphs not only provide better worst-case hop counts, thus improving performance, but also guarantee packet delivery. The intuition behind these graphs is that when a packet misroutes due to an off link along its minimal path[2], it will only be one hop away from that minimal path. Hence, considering just the incoming port of a packet, each router has sufficient knowledge to quickly steer the packet back to the minimal route. Our proposed on-demand graph, however, does not guarantee absolute worst case hop counts as any link can potentially turn off, allowing network connectivity to take any non-deterministic form. Thus the worst-case hop count can theoretically be infinite. Therefore, this graph should only be applied with relatively fast on/off links, so quick re-connections among adjacent routers can be established allowing packets to route to their destinations in a prompt fashion (when governed by a fully adaptive deadlock-free routing function). In the next sections we will elaborate on such a design.

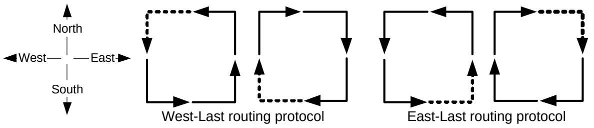

Fig. 8. Turns permitted (shown with solid lines) and disallowed (shown with dashed lines) for the West-Last and East-Last routing protocols.

B. How to Route Packets when Links are Off?

After defining the on/off connectivity graph, the designer needs to derive a deadlock-free on/off routing protocol - a protocol that has to deliver packets irrespective of the state of on/off link candidates during network operation.

While a trivial deadlock-free protocol can strictly use ever-on links, that will hurt network performance severely. An on/off routing protocol needs to be able to leverage the bandwidth of on/off link candidates when they are active, steer traffic away from them when they are shut down, while always guaranteeing deadlock freedom.

Here, we derive routing algorithms matched to each of the three connectivity graphs.

*1) **Alternating Row-Column – West-Last East-Last Routing**:* West-last (east-last) routing, as the name suggests, restricts routes so turns from the west direction (east) are disallowed, as shown in Fig. 8. While it is well-known that such protocols that are based on the Turn Model [14] ensure deadlock freedom in meshes and tori [15], for it to be an on/off routing protocol, it needs to be deadlock-free with the ever-changing and limited connectivity of an on/off network.

To ensure deadlock freedom with the alternating row-column connectivity graph, we split the network into two virtual networks, one governed by non-minimal west-last (WL) and the other by non-minimal east-last (EL) routing protocols. A virtual network is a superimposed map of the entire physical network which holds a logical state, with a virtual channel mapped onto a specific link (holding the state of that link). In a mesh, packets that route eastwards use the WL virtual network while those that route westwards use the EL virtual network. In a torus, packets utilize EL when travelling anticlockwise and WL when going clockwise. Packets do not switch between virtual networks once they are injected, thus the entire network is deadlock-free so long as each virtual network is deadlock-free. Intuitively, the ever-on links towards the west (east) direction in an alternating row-column connectivity graph ensure that the WL (EL) virtual network will not have cycles, regardless of the state of on/off link candidates. A detailed mathematical proof of deadlock and livelock freedom is available at [13], and omitted here for brevity. Note that due to the symmetry of the alternating row-column graph, a combination of south-last and north-last routing protocols can also be used.

*2) **Staircase – Non-minimal Optimal-Y (Opt-Y) Routing**:* An analysis of all possible routing scenarios of the staircase connectivity graph, across all possible on/off link states, shows that a routing protocol coupled to this graph must permit complete cycles while ensuring deadlock freedom. We therefore devise a non-minimal version of the fully-adaptive Opt-Y [16] routing function with our proposed staircase connectivity graph for both meshes and tori (though originally proposed in minimal form for meshes) that allows this cyclic behavior.

With this routing function no virtual networks are needed. A packet routes progressively towards its destination until it hits an off link. It then misroutes for a hop; this direction can still be progressive due to the staircase nature of the graph.

*3) **On-demand – Any Deadlock-Free Fully-Adaptive Routing**:* In an on-demand graph, since any link and any number of links in a router can turn off, it is apparent that for deadlock freedom, a fully

[2] A minimal path is a path with the shortest number of hops between the source and destination.

adaptive routing algorithm in which traffic can form cycles without inducing a deadlock is needed. Any deadlock-free fully-adaptive routing protocol, such as Opt-Y, will work.

C. When to Put a Link to Sleep?

Given an on/off connectivity graph, the sleep mechanism determines when to shut down the on/off link candidates.

Since links have on/off transition delays that exceed the typical router pipeline delay, a naïve per cycle approach where a link is put to sleep whenever it is unused would induce severe performance degradation. Instead, the sleep mechanism needs to base its decisions on statistics gathered and compared against thresholds.

Statistics. Statistics can be gathered per-port, i.e. sleep decisions are made based only on local information at each port; or per-router, i.e., sleep decisions are made collectively based on statistics gathered across all ports in a router.

Link utilization is the most direct per-port statistic – it tracks the percentage of time a link is idle. This statistic tracks resource utilization well at low to mid network traffic levels; however when the network is congested traffic tends to get buffered in input and output buffers and link utilization leans to zero, making it an unsuitable metric under such a condition. Consequently per-port buffer utilization (the buffer attached to a link) can measure network traffic levels by indicating the number of buffer entries occupied per unit duration of time. Lastly, per-port buffer age measures the duration of time flits remain in associated buffers until they depart the current link towards the next router or destination.

In an analogous manner, statistics that can be used to track network utilization on a per-router mode are: (1) aggregate input buffer utilization, U_{buffer}, the total number of occupied input buffers, (2) aggregate output link utilization, U_{link}, the number of cycles when output links are in use and (3) aggregate input buffer age, Age_{buffer}, the time flits spend waiting in input buffers:

$$U_{buffer} = \frac{\sum_{p_{in}=1}^{P_{in}} \sum_{t=1}^{T_{sw}} (F(t, p_{in})/B)}{T_{sw} \times P_{in}}, \quad 0 \leq U_{buffer} \leq 1 \tag{1}$$

$$U_{link} = \frac{\sum_{p_{out}=1}^{P_{out}} \sum_{t=1}^{T_{sw}} A(t, p_{out})}{T_{sw} \times P_{out}}, \quad 0 \leq U_{link} \leq 1 \tag{2}$$

$$\text{where} A(t, p_{out}) = \begin{cases} 1 & \text{if traffi c passes link } p_{out} \text{ in cycle } t \\ 0 & \text{if no traffi c passes link } p_{out} \text{ in cycle } t \end{cases}$$

$$Age_{buffer} = \frac{\sum_{t=1}^{T_{sw}} \sum_{i=1}^{\sum_{p_{in}=1}^{p_{in}=4} d(p_{in},t)} (t_{d_i} - t_{a_i})}{\sum_{t=1}^{T_{sw}} \sum_{p_{in}=1}^{p_{in}=4} d(p_{in}, t)} \tag{3}$$

where $F(t, p_{in})$ is the number of input buffers occupied at time t for active input ports p_{in} and B is the input buffer size. P_{in} and P_{out} are the number of active input and output ports respectively in a router. $d(p_{in}, t)$ is a flit arriving at time t_{a_i} and departing at time t_{d_i} at input buffer p_{in}. T_{sw} is the sampling window size in terms of system clock cycles.

Another alternative is the weighted combination of statistics so multiple resources can be considered in tandem. For instance predicted aggregate input buffer utilization $\Pi_{buffer_{predict}}$ can be combined with predicted aggregate link utilization $\Pi_{buffer_{predict}}$ by weighting each statistic, where $0 \leq \beta \leq 1$:

$$0 \leq \Pi_{stat_{predict}} = (\Pi_{buffer_{predict}} \times \beta) + (\Pi_{link_{predict}} \times (1 - \beta)) \leq 1 \tag{4}$$

Statistics Smoothing and Prediction. It is important to be able to distinguish between short-term variations in traffic from more stable trends in traffic patterns. Otherwise, network congestion may occur when traffic picks up after a temporary dip, as a link that is turned off can not be immediately reactivated.

An effective prediction mechanism is exponential weighted average: It combines past (obtained during the last sampling period T_{sw}), Π_{past}, and current, $\Pi_{current}$, statistics, smoothing and predicting future traffic, $\Pi_{predict}$:

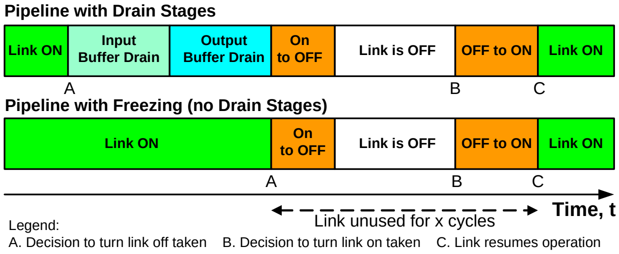

Fig. 9. Router pipelines with (a) flits drained from the router vs. (b) flits frozen in place.

$$\Pi_{predict} = \frac{\Pi_{current} \times weight + \Pi_{past}}{weight + 1} \tag{5}$$

It is critical to keep in mind the hardware overhead involved in gathering statistics. For all statistics we propose, only simple hardware counters are needed[3]. The weights ($weight$ set equal to 3 in our experiments) of the prediction mechanism are also chosen so a shifter (for division) can be used.

Thresholds. The simplest threshold is a single threshold α_{sleep}, so when the statistics $\pi_{stat_{predict}}$ falls below it, the link is put to sleep.

Multiple thresholds can also be set, corresponding to the number of active links, so the more links that are asleep, the harder it is to turn further links off. For instance, a router with four links will have four thresholds α_{sleep_1} to α_{sleep_4}, where α_{sleep_4} is used when all four links are on, α_{sleep_3} is used when 3 are on, and so on, and α_{sleep_1} triggers at a lower activity level than α_{sleep_2} and so on.

In addition, thresholds can be fixed at design-time, or adaptive, to run-time traffic characteristics. Adjustable thresholds allow us to adapt the sleep decision – when it introduces too much delay, the threshold can be tightened. We propose exponentially backing off the thresholds (i.e., doubling/halving the thresholds accordingly) whenever sleep decisions introduce excess latency. This is similar to the exponential-backoff collision-avoidance mechanism in Ethernet. Here, aggregate buffer age statistics are gathered, smoothed and used to predict the delay introduced by the sleep decision.

D. When to Wake Up a Link?

The decision to turn a link on can be on-demand – i.e., whenever a flit demands that link, start reviving it right away (though it won't be active till after the off-to-on delay).

Alternatively, the wakeup decision can similarly be based upon statistics and thresholds. The design space here is then similar to that for the sleep decision and is omitted here for brevity.

E. How to Implement On/Off Within the Router Pipeline?

In incorporating on/off within the router pipeline, the designer needs to decide if the pipeline should be frozen upon a sleep decision, and restarted at wakeup, or if the packets currently in the pipeline should be drained. Fig. 9 shows the router pipeline for the two cases. Draining introduces more pipeline delays for the current packet flow but lowers the possibilities of deadlocks. Freezing can cause delays for other flows as well as deadlocks as a frozen packet holds both current and downstream router resources, severely limiting the resources of other packets.

In a pipeline incorporating buffer drain stages, once the tail flit of the last packet departs from the input buffer, it is wise to wait till the output buffer coupled to the link to be turned off is free of flits as well, so remnants of a packet can completely leave the current router.

[3] The power consumed by counters is ignored, as similar hardware has been shown to consume little CMOS area with negligible power [17].

IV. Design Case Studies

The alternatives proposed and explored at each step of our 5-step design methodology in Section III can be combined to form numerous power-aware on/off network designs. Here, we present in detail two design case studies – One targets slow on/off links that take in the order of 10,000 cycles to turn on/off, and another focuses on links that can rapidly be switched on/off, in the order of $< 1,000$ cycles. Through the two case studies, we seek to show how our methodology and design space helps a designer pick designs with vastly different latency versus power savings tradeoffs.

A. Case Study I: Slow On/Off Links

The alternatives explored in this case study at each step of our methodology are:

- Which links to turn off?
  - Alternating Row-Column
  - Staircase
- How to route packets when links are off?
  - WLEL
  - Opt-Y
- When to put a link to sleep?
  - Statistics: Per-router combination of U_{buffer} and U_{link}
  - Thresholds: Multiple, fixed
- When to wake up a link?
  - Statistics: Per-router combination of U_{buffer} and U_{link}
  - Thresholds: Multiple, fixed
- How to implement on/off within a router pipeline?
  - Draining

With on/off link transitions taking in the order of 10,000 cycles, disconnectivity and deadlocks are of prime concern. An on-demand connectivity graph is thus not feasible, as it can lead to infinite delays, as shown in Table I. We explore the alternating row-column and staircase graphs in this case study, along with their companion routing algorithms WLEL and OptY respectively that ensure deadlock freedom.

For the sleep mechanism, per-port statistics are not compatible with the selected connectivity graphs since we allow two on/off link candidates per router, so not all of the four links in a router can be turned off. We thus choose per-router aggregate statistics, and settle on a combination of U_{buffer} and U_{link} of Equation 4, with $\beta = 0.7$ as U_{buffer} is found to be more indicative of network resource contention than U_{link}.

Fixed, multiple thresholds are chosen, with $\alpha_{sleep_1} = 2\%$, and $\alpha_{sleep_2} = \alpha_{sleep_1} + \delta_{sleep}$ and so on ($\delta_{sleep} = 2\%$). These thresholds were chosen as the small packet sizes assumed (5-flit packets) lead to buffers in virtual-channel routers being fairly under-utilized. By choosing threshold levels that are reflective of network traffic, we can avoid false triggering and achieve good network power-performance.

For wakeup, turning the link on each time a flit demands it is not practical given the long on/off delay. We thus use analogous statistics and thresholds as in the sleep mechanism, with $\alpha_{wake_1} = 15\%$ (threshold when one link is off) and $\delta_{wake} = 2\%$. For wakeup, we subtract an additional δ_{wake} when an extra link is off, tightening our thresholds so that they can present more sensitive responses to abrupt increases in traffic with fewer links available (on) to handle this extra load.

Draining is chosen over freezing in the router pipeline implementation since the methodology outlines the severe blockage downstream that can occur with slow on/off links.

B. Case Study II: Fast On/Off Links

The alternatives explored here are:

- Which links to turn off?
  - On-demand
- How to route packets when links are off?
  - Fully-adaptive routing – OptY
- When to put a link to sleep?
  - Statistics: Per-port link utilization
  - Thresholds: Multiple, adaptive (exponential backoff)
- When to wake up a link?
  - On-demand
- How to implement on/off within a router pipeline?
  - Freezing

With on/off links that have faster on/off responses in the order of 100s of cycles [9], the full connectivity presented in Section IV-A is no longer a crippling concern. In this case study we design networks for fast on/off links using an "on-demand" scheme in which inactive links turn on when a flit demands use of them and similarly go to sleep after a period of inactivity (the time elapsed since a link was last traversed even by a single flit; a future flit departing from this same link would reset that link's inactivity count).

So, an on-demand connectivity graph along with Opt-Y routing is selected. For sleep, multiple thresholds are used, with $\alpha_{sleep_4} = 4 \times \alpha_{sleep_3}$ and so on. These four thresholds are also adaptive, exponentially backing off (doubling their levels) whenever aggregate buffer age Age_{buffer} exceeds a user-defined preset tolerance value, and return to their original values when Age_{buffer} is within the specified target. In our experiments we set this tolerance value equal to a $25\% Age_{buffer}$ overshoot; Increasing this number makes the sleep thresholds less adaptive to sudden increases in traffic levels therefore prolonging link sleep durations, incurring greater power savings at the expense of a higher network latency. Decreasing the Age_{buffer} tolerance level induces the opposite effect on achievable power savings and network latency levels. Wakeup is on-demand (when even a single flit demands use of an inactive link) since the link can be turned on quickly. While the link is waking up, the routing algorithm re-routes adaptively using Opt-Y so traffic is not stalled throughout the wakeup time.

V. Evaluation of Case Studies

A. Simulator Setup

We implemented an event-driven, flit-level interconnection network simulator with credit-based flow control extending upon a publicly available simulator PopNet [18] that covers both designs described in our case studies of Section IV. The simulator supports k-ary 2-cube topologies consisting of 5-stage pipelined virtual-channel routers – (1) routing, (2) virtual channel arbitration, (3) switch allocation, (4) switch traversal and (5) output link traversal.

We assumed 1GHz routers, each with 2 virtual channels, 40 flit input and output buffers. Packets consist of five 32-bit flits with each flit transported in 1 cycle over 32Gb/s links. Each router consists of 8 unidirectional channels (four incoming and four outgoing).

Each simulation is run for 10 million cycles in the off-chip networks and for the entire trace length for the on-chip networks. The metrics considered are latency, throughput and power consumption. Latency spans the injection of the head flit of a packet until its tail flit is ejected from the destination router. The saturation throughput of the network is considered to be the point where the average latency of the packets is double the zero-load latency. Power savings is the ratio of the aggregate power savings across all links in the network, divided by the power consumption if all links are operational. Router power is ignored since our approach will only increase congestion and corresponding arbitrations, but arbitration power has been shown to be an insignificant portion of total router power [19].

B. Benchmarks

1) ***On-Chip Benchmarks****:* To better evaluate the effectiveness of our proposed on/off network, we obtained network traces from the TRIPS chip-multiprocessor [6]. The TRIPS chip-multiprocessor consists of 4 large, coarse-grained element cores each of which is an instantiation of the Grid Processor Architecture (GPA) containing an ALU execution array and local L1 memory tiles interconnected via a 5 × 5 network.

TRIPS network packets carry data (operands for instructions or addresses to memory) and status information associated with them. We assumed packets consisting of three 32-bit flits and 1GHz router cores. Network traces were obtained from simulations of a suite of thirteen SPEC2000 and MediaBench [10] benchmarks. The traces are in general very bursty – large numbers of packets injected at times, and zero packets at others, which present interesting opportunities for power optimization. The traces also exhibit high spatial variance giving good opportunities to optimize power by rerouting packets around inactive links without impacting throughput.

*2) **Off-Chip Benchmarks**:* We used a synthetic workload model proposed in [17] that exhibits both high temporal and spatial variance, a reflection of real-world communication traffic. The workload consists of a two-level self-similar or long-range dependence (LRD) communication traffic workload. On the first level, the network traffic consists of communication tasks that are generated at random nodes based on the model of sphere of locality [20], with Poisson inter-arrival times. The tasks are assumed to have an average duration of 1μs. At the second level, within each task session, packet arrivals are self-similar, generated by multiplexing ON/OFF sources that have Pareto-distributed ON/OFF periods. Packets are 5 flits long.

Four sets of self-similar traffic with 10, 25, 50 and 100 concurrent tasks were simulated, with workloads with fewer tasks exhibiting greater temporal and spatial traffic level variability. We therefore expect to see more opportunities for power optimization for workloads with a smaller number of concurrent tasks.

C. On/Off Link Model

The performance and efficiency of an on/off link is limited by the inherent on to off (and off to on) state transition delay, t_{trans}. A longer link shutoff reduces the potential of power savings, as power is also consumed during a transition. Longer reactivation times limit the potential for power savings since packets need to stall longer at a router, waiting for the link to be up.

In our experiments, we assumed a conservative link power model – that the power consumption of a link stays constant when it is on, since link worst-case and nominal power levels are close (the IBM InfiniBand 12X LPE TX consumes a nominal power of 0.26W and a worst case power of 0.3W while its RX takes up nominal 0.17W and worst case 0.2W); that a link takes zero power when shut off; and consumes the same power as when it is on during input and output buffer draining and transitions from on to off (sleep) states and vice versa (wake).

As on-chip links are able to leverage the low-skew global chip clock as the clocking source at transmitters and receivers [8] [9], link calibration time when transitioning from off to on states is likely to be substantially shorter than that in board-to-board links. Preliminary measurements by Shepard et. al. on their 8Gbps on-chip serial link that uses differential, pulsed current-mode signalling [9] exhibit the following behavior: (a) When drivers, receivers and DLLs of the link are shut off, the link can be back in operational state after 200 cycles, with 75% power savings when the link is off; (b) When only drivers and receivers are turned off, a rapid on/off transition delay of 2 cycles can be attained, but with a lowered 25% power savings during off time since DLL and seriailizing/deserializing flip-flops take up about 75% power.

We thus evaluate our Design II that is targeted for fast on/off links with on-chip network traffic traces, while Design I is evaluated against the synthetic self-similar traffic traces that are representative of the traffic in blade cluster systems with slow board-to-board on/off links.

It should be noted that model (b) enables the maximum potential of a naïve on-demand policy where a link is turned on whenever it is going to be used, and sleeps otherwise, since router pipelines are typically longer than 2 cycles. Its power savings of 25% thus represents the best a naïve on/off network can achieve, assuming perfect oracle knowledge of traffic patterns. As will be shown later in our case studies, our schemes can surpass those power savings, assuming the conservative model that was also applied to our off-chip experiments as described above.

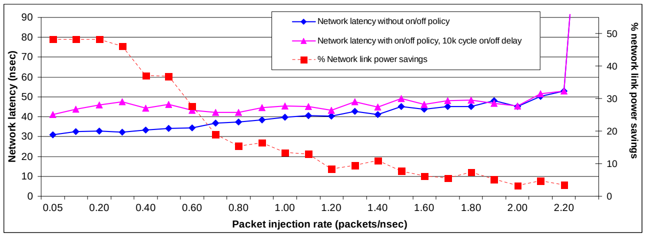

Fig. 10. Power savings and latency-throughput of an 8×8 mesh staircase connectivity graph for 100 concurrent tasks with $t_{trans} = 10$k cycles.

D. Case Study I: Full-Connectivity Graphs for Slow On/Off Links

Fig. 10 shows the power-performance of an 8×8 mesh topology with self-similar traffic of 100 concurrent tasks, incorporating the staircase connectivity graph with Opt-Y routing. It shows our design effectively adapting to varying injection rates – at low injection rates, power savings reach the maximum of 50%, since network utilization is low, resulting in the run-time decision mechanism turning many links off to save power. At this range the latency penalty is relatively higher as packets traverse longer paths to their destinations increasing the hop count. At higher packet injection rates, power savings with on/off links are reduced, as there exist much fewer opportunities for turning off links. Interestingly, at 1.90 packets per cycle injection rate, our power-aware design actually leads to an improvement in latency as it better balances traffic.

Table II gives a summary for all experiments carried out, covering both proposed connectivity graphs, mesh and torus topologies and different number of concurrent tasks. The average values reported here are obtained by averaging the statistics from zero-load to just before network saturation. The trends are:

- Faster on/off link delays (1k cycles) give rise to better power savings and lower latency penalties.
- The staircase connectivity graph (Opt-Y routing) gives better performance than the corresponding metrics of the alternating row-column connectivity graph due to lower average and worst case hop counts as links turn on/off requiring packet rerouting.
- The staircase connectivity graph also produces higher power savings due to the greater number of on/off link candidates in the case of 2-ary mesh networks.
- The 50 concurrent task workloads give better power savings as opposed to the 100 task workloads due to the better opportunities for power savings as traffic exposes greater variance to the on/off mechanism and on/off links.

Note that the power savings are substantially greater than the near-zero theoretical results attainable with an oracle (Section II). This power savings is not without cost though – coming at a noticeable latency penalty due to the heavy re-routing needed with so many links off. However, in many board-to-board systems such as blade clusters, the network interface delay is in the order of 100s of cycles [12], and hence easily mask the 4 to 5 additional cycles introduced on average by the power-aware on/off network.

TABLE II
POWER-PERFORMANCE OF OUR PROPOSED ON/OFF APPROACHES FOR 8-ARY 2-CUBE TOPOLOGIES WITH SELF-SIMILAR TRAFFIC.

Topology	# Tasks	Metric	$t_{trans}(link)$ (cycles)			
			Alt. Row-Column		Staircase	
			$1,000$	$10,000$	$1,000$	$10,000$
Mesh	50	Latency penalty	25.25%	27.31%	25.26%	25.33%
		Power savings	23.16%	24.70%	31.55%	29.98%
Torus	50	Latency penalty	32.63%	36.82%	22.03%	22.57%
		Power savings	33.55%	33.23%	28.81%	28.60%
Mesh	100	Latency penalty	15.54%	20.77%	16.01%	16.57%
		Power savings	13.54%	14.04%	19.02%	19.17%
Torus	100	Latency penalty	25.21%	27.12%	15.07%	16.36%
		Power savings	23.63%	24.26%	19.05%	19.26%

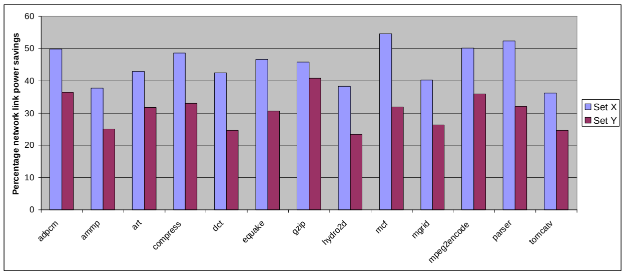

Fig. 11. Percentage total network link power savings for a 10 cycle on/off link transition delay.

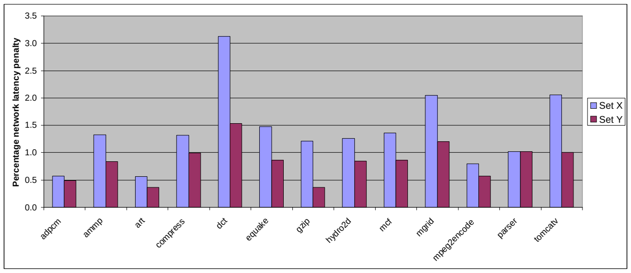

Fig. 12. Percentage increase in network latency for a 10 cycle link on/off transition delay.

E. Case Study II: On-Demand Connectivity Graphs with Adaptive Inactivity Thresholds

We run experiments using two different sets of thresholds as shown in Table III for our real TRIPS on-chip network traffic traces. Figs. 11 and 12 show the percentage total network link power savings and the corresponding network latency penalty for a link transition delay of 10 cycles for the TRIPS CMP running a suite of 13 benchmarks for MediaBench [10] and SPEC2000 applications.

Table IV supplies the rest of the results for 100 and $1,000$ cycle link on/off transition delays. Here, the fast on/off links enabled substantial power savings of up to 54.4% with an insignificant latency penalty that is kept under 7.5% even with a 1,000-cycle on/off link delay. Note that the power savings achieved here for the $1,000$ cycle on/off link delay are considerably greater than those achievable by an oracle in Section II-B; As mentioned, those results are under ideal conditions where the performance penalty incurred is zero, while here we incur a minute increase in delay for even greater power savings. The tighter threshold levels of set Y produce relatively lower power savings, albeit at smaller latency penalties too. Table V provides the overall power savings and latency increase percentages across the entire suite of benchmarks tested.

TABLE III
SETS OF MINIMAL ADAPTIVE BACK-OFF INACTIVITY THRESHOLDS.

Clock cycles	Set X	Set Y
All 4 links on, α_{sleep_1}	1,000	1,600
3 links on, 1 off, α_{sleep_2}	4,000	6,400
2 links on, 2 off, α_{sleep_3}	16,000	25,600
1 link on, 3 off, α_{sleep_4}	64,000	102,400

How do the various benchmarks behave? We have analyzed the various benchmarks used here and we found that they exhibit different temporal characteristics. Benchmarks such as `tomcatv` have frequent, large bursts of packets injected. In this case, opportunities to turn off links are fewer leading to lower power savings. A slower on/off link also causes a relatively higher latency penalty as it cannot turn back on before the next large burst of traffic, thus causing many packets to be re-routed. On the other extreme, benchmarks with infrequent and small bursts of traffic such as `ammp` present greater opportunities for power optimization. We see the multiple, adaptive thresholds here effectively tracking the different characteristics of the benchmarks, fine-tuning the on/off network effectively.

How do results here compare with those of Case Study I? Considerably higher power savings along with a smaller impact on latency were achieved with the on/off interconnection network design presented in Case Study II as compared to the network design described in Case Study I. The various reasons that explain this observance are listed here:

- Substantially slower links of up to 2 orders of magnitude (for self-similar traffic) were considered in case study I that are more representative of off-chip box-to-box and chip-to-chip interconnects. Slower on/off responses translate to greater performance penalties and smaller potential for power optimizations as power is also burned during link state transitioning.
- The graphs of Case Study I had to overcome the challenge of network disconnectivity in conjunction with a slow on/off link response. This combination translates to heavier packet rerouting than in the on-demand graph; since packets traverse longer paths when links are off, the average latency of the network increases.
- Maximal power savings for the alternating row-column and staircase graphs are limited to an average of 50% as each design provides a maximum of two *on/off link candidates* out of a pool of 4 links at every network router (the two other links are *ever-on* links). The on-demand graph, however, does not present this restriction as every link can potentially turn on/off (every link is an *on/off link candidate*), leading to theoretical (though unrealistic) maximal power savings of 100% with all links off. As a consequence, in our experiments under Case Study I we see power savings below the 50% mark, while in Case Study II network link power savings were seen to surpass the 50% line in some of the experiments.

TABLE IV
PERCENTAGE POWER SAVINGS AND LATENCY INCREASE FOR 100 AND $1,000$ CYCLE ON/OFF LINK TRANSITION DELAYS WITH ON-CHIP NETWORK TRACES.

	Power Savings/Latency (%)							
On/off delay	100 cycles				$1k$ cycles			
Threshold Set	X		Y		X		Y	
`adpcm`	49.8	0.7	36.3	0.6	50.6	3.1	34.9	2.3
`ammp`	46.9	1.5	25	0.8	40.2	3.8	24.7	1.4
`art`	42.6	0.9	30.3	0.5	52.9	2	32.9	1.8
`compress`	49.1	2.4	33.3	1.1	45.6	4.5	54.4	3.5
`dct`	38.7	2.6	26.1	2.1	41.2	0.4	44.7	2.4
`equake`	43.2	2.7	29.1	1.2	50.3	4.7	28.9	0.6
`gzip`	50.6	1.3	39.4	0.3	47.4	3	47.8	3.2
`hydro2d`	41.6	1.4	26.9	1	44.9	2.9	27.6	1.3
`mcf`	52	1.6	29.9	1.2	41.6	1.3	33.2	3
`mgrid`	36.9	1.8	24.6	1.2	53.9	6	42.3	5
`mpeg2encode`	47.2	1.3	32.5	0.6	49.1	0.5	32.6	1.1
`parser`	53	1.2	34.4	1.7	50.6	7.5	30	6.4
`tomcatv`	44.5	1.9	23.8	0.9	39.8	5.7	30.7	2.8

TABLE V
AVERAGE POWER SAVINGS AND LATENCY PENALTY FOR THE ENTIRE CMP BENCHMARKS SUITE.

	Power Savings/Latency (%)			
Threshold Set	X		Y	
10 cycle on/off delay	45.1	1.4	30.5	0.8
100 cycle on/off delay	45.9	1.6	30.1	1
$1k$ cycle on/off delay	46.7	3.5	35.5	2.7

VI. RELEVANT WORK

Power-aware networks were first proposed recently [21] [17]. Previously, designers only had the option of using lower frequency links throughout the network slowing all network traffic across the board, unless they had available accurate prior knowledge of the applications running on the network. The first power-aware network proposed explored dynamic voltage scalable (DVS) links [17] that scale link frequency and voltage to track utilization. Thereafter, networks with DVS-DLS (dynamic link shutdown) [22] links that further shut down DVS links when traffic drops to very low levels were proposed and investigated. In comparison to DVS and DVS-DLS links, plain on/off links require much simpler hardware and are already commercially available [11], albeit with 10s of thousands of cycles on/off delays. DVS and DVS-DLS links, on the other hand, require substantial circuits innovations to extend variable-frequency links [23] to support fast, voltage and frequency changes while ensuring correct link operation during voltage scaling, thus incurring higher power and area overhead. Clearly, the power savings realizable with an on/off link is also greater as compared to DVS links that still consume significant power while idle.

Building power-aware networks with on/off links present tougher challenges though. The challenges of overcoming network disconnectivity and potential network deadlocks, and re-routing when links are asleep are not faced with DVS links, since the network remains always connected. With DVS-DLS links, network disconnectivity is not a major problem as the researchers assumed fairly optimistic link on/off times of 800ns and the targeted system is clusters where sensitivity to delay is not as high, so the delay introduced by the proposed run-time route reconfiguration approach which updates all affected routing tables upon each on/off decision is still manageable.

Our approach bears similarities to prior work in fault-tolerant routing protocols and theories [15] [24], steering network traffic when network connectivity is reduced as links shut down to save power. However, fault-tolerant approaches are *reactive* and merely performance oriented – they respond to link and node failures (specific algorithms handle just a single network link failure) when they occur, either delivering packets to their destinations or signalling regions of failures when connectivity is not feasible. Instead, our on/off link policy is *proactive*, exploiting the advance knowledge of link on/off state that is unavailable for fault-tolerant routing protocols, methodically steering traffic for best power-performance. It should also be noted that fault-tolerant protocols typically target one or a handful of faults, while in an on/off network, substantial number of links need to be put to sleep to realize power savings.

In this paper, we target hardware power management of on/off networks, where an up-to-date global view of the on/off network is just not possible. An interesting avenue for future research is software-based approaches, such as where compilers can leverage their global view of network utilization to better power-performance of on/off networks.

VII. Conclusions

Network power is becoming a barrier to system scalability and needs to be critically addressed. This paper explores the design space of power-aware on/off networks, proposing a 5-step design methodology to guide designers through the challenges of maintaining good overall network power-performance. At each step of our methodology, we propose several solutions that can then be assembled in interesting ways to form the blueprint of on/off network designs. We then describe and evaluate two specific designs targeted to off-chip and on-chip networks with varying on/off link delays. With on/off links already commercially available, we hope our methodology will pave the way for the deployment of power-aware networks.

Acknowledgments

The authors would like to thank Ken Shepard, Anup Jose and George Patounakis of Columbia University for explaining design constraints of on/off links as well as providing access to actual link measurements, the TRIPS Team at the University of Texas at Austin for supplying us with the traffic traces of various benchmarks, Edward Lee of Velio Communications and Gu-Yeon Wei of Harvard University for insightful technical discussions on feasibility and constraints of on/off links, and Li Shang of Princeton University for support of his PopNet network simulator. This work was supported in part by the National Science Foundation (NSF) under contract number CCR-0237540 (CAREER) and the MARCO Gigascale Systems Research Center.

References

[1] S. S. Mukherjee, P. Bannon, S. Lang, A. Spink, and D. Webb, “The Alpha 21364 network architecture,” *IEEE Micro*, vol. 22, no. 1, pp. 26–35, Feb. 2002.

[2] W. J. Dally, P. Carvey, and L. Dennison, “The Avici terabit switch/router,” in *Proc. Hot Interconnects 6*, Aug. 1998.

[3] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W. Su, “Myrinet - a gigabit-per second local-area-network,” *IEEE Micro*, vol. 15, no. 1, pp. 29–36, Feb. 1995.

[4] “InfiniBand Trade Alliance, the InfiniBand architecture.” [Online]. Available: http://www.infinibandta.org/

[5] W. J. Dally and B. Towles, “Route packets, not wires: On-chip interconnection networks,” in *Proc. Design Automation Conference*, June 2001, pp. 684–689.

[6] 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. of the 30th International Symposium on Computer Architecture*, June 2003, pp. 422–433.

[7] 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*, Aug. 2003, pp. 424–427.

[8] R. Ho, K. Mai, and M. Horowitz, “Efficient on-chip global interconnects,” in *Symposium on VLSI Circuits*, June 2003, pp. 271– 274.

[9] A. Jose, G. Patounakis, and K. Shepard, “A 8Gbps on-chip serial link,” Columbia University, Tech. Rep., TR10-03-01, Nov. 2003.

[10] C. Lee, M. Potkonjak, and W. H. Mangione-Smith, “MediaBench: A tool for evaluating and synthesizing multimedia and communications systems,” in *Proc. of the 30th International Symposium on Microarchitecture*, Dec. 1997, pp. 330–337.

[11] “Motorola inc.” [Online]. Available: http://www.motorola.com/

[12] V. Karamcheti and A. A. Chien, “Software overhead in messaging layers: Where does the time go?” in *Proc. ASPLOS VI, San Jose, CA*, Oct. 1994, pp. 51–60.

[13] V. Soteriou, “Proof of deadlock-freedom for west-last east-last routing for on/off interconnection networks,” Princeton University, Tech. Rep., TR-06-04, June 2004.

[14] C. J. Glass and L. M. Ni, “The turn model for adaptive routing,” in *19th International Symposium on Computer Architecture,*, May 1992, pp. 278–287.

[15] J. Duato, “A theory of fault-tolerant routing in wormhole networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 8, no. 8, pp. 790–802, Aug. 1997.

[16] L. Schwiebert and D. N. Jayasimha, “Fully adaptive wormhole routing for meshes,” in *Proc. of Supercomputing*, Nov. 1993, pp. 782–791.

[17] L. Shang, L. Peh, and N. K. Jha, “Dynamic voltage scaling with links for power optimization of interconnection networks,” in *Proc. of the 9th International Symposium on High-Performance Computer Architecture (HPCA)*, Feb. 2003, pp. 79–90.

[18] “Popnet simulator.” [Online]. Available: http://www.ee.princeton.edu/edu/~lshang/popnet.html

[19] H. Wang, X. Zhu, L. Peh, and S. Malik, “Orion: A power-performance simulator for interconnection networks,” in *Proceedings of the 35th International Symposium on Microarchitecture (MICRO)*, Nov. 2002, pp. 294–305.

[20] D. A. Reed and D. C. Grunwald, “The performance of multicomputer interconnection networks,” *IEEE Computer*, vol. 20, no. 6, pp. 63–73, June 1987.

[21] L. Benini and G. D. Micheli, “Powering networks on chips: Energy-efficient and reliable interconnect design for SoCs,” in *Proc. International Symposium on Systems Synthesis*, Oct. 2001, pp. 33–38.

[22] E. J. Kim, K. H. Yum, G. M. Link, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, M. Yousif, and C. R. Das, “Energy optimization techniques in cluster interconnects,” in *International Symposium on Low Power Electronics and Design*, Aug. 2003, pp. 459–464.

[23] G. Wei, J. Kim, D. Liu, S. Sidiropoulos, and M. Horowitz, “A variable-frequency parallel I/O interface with adaptive power-supply regulation,” *J. Solid-State Circuits*, vol. 35, no. 11, pp. 1600–1610, Nov. 2000.

[24] P. T. Gaughan and S. Yalamanchili, “A family of fault-tolerant routing protocols for direct multiprocessor networks,” *IEEE Trans. on Parallel and Distributed Systems*, vol. 6, no. 5, pp. 482–497, May 1995.