Cubic Ring Networks: A Polymorphic Topology for Network-on-Chip

Bilal Zafar and Jeff Draper
Ming Hsieh Department of Electrical Engineering
Information Sciences Institute
University of Southern California
Marina Del Rey, CA 90292
Email: {bzafar,draper}@isi.edu

Timothy M. Pinkston
Ming Hsieh Department of Electrical Engineering
University of Southern California
Los Angeles, CA 90089
Email: tpink@usc.edu

***Abstract*—As chip multiprocessors transition from multi-core to many-core, on-chip network power is increasingly becoming a key barrier to scalability. Studies have shown that on-chip networks can consume up to 36% of the total chip power, while analysis of network traffic reveals that for extended periods of execution time, network load is well below the network capacity in many applications. In recent studies, researchers have proposed to exploit this temporal variability in network traffic to dynamically turn off links, buffers and segments of the on-chip routers. In this work, we make the case for a polymorphic topology, called Cubic Ring (cRing), that allows dynamically turning off over 30% of resources in a 2D network (and more in higher dimensional networks), with less than 5% increase in average distance. As a result, cRing networks provide an elegant way to trade off network bandwidth for lower (static) power. A complete formalism for the proposed cRing topologies and the associated routing algorithm is presented, along with evaluation under synthetic workloads.**

I. Introduction

The opportunities afforded by Moore's Law - doubling transistor density with each process generation - are increasingly challenged by three problems: on-die power dissipation, wire delays and design complexity [1]. The core-based architecture approach to designing chip multiprocessors (CMPs) proposes to alleviate these problems by dividing each die into a modest number of low-power cores which are connected via a packet-switched fabric. This modular approach enables rapid scaling in the number of cores implemented on a die with 10s of cores emerging [2] and 100s of cores expected within a decade [3].

When the core-count grows to tens of cores on a die, the on-chip network becomes a potential bottleneck from both performance and power standpoints. An *ideal* on-chip network should consume power only when it is delivering packets, not when it is waiting for packets to be injected. More precisely, only those network resources (links, buffers, crossbar, etc.) that are along the path of packets currently in the network should consume power, while the rest should be *off*[1]. However, the state-of-the-art in on-chip networks is very different from this ideal model. On-chip networks in implemented designs are always ready to operate at their peak performance and consume a significant portion of the total chip power (40% and 28% of each core's power budget in MIT Raw [8][9] and Intel Teraflop [10] processors, respectively). Given tight power constraints, there is a growing need for power-efficient on-chip networks with sufficient bandwidth. In [10], for example, in order to prevent the network from becoming a bottleneck, it was argued that CMPs fabricated in 32nm must be able to provide bisection bandwidth of more than 2 TeraBytes/sec and consume under 10% of the total chip power.

A potential approach for meeting power-performance goals is to turn *off* network links, ports and parts of the router when the traffic injection rate is low, and turn them back *on* only when demand for bandwidth increases sufficiently. The key observation behind this approach is the temporal variance in network traffic injection rate across applications. In [11], it was shown that in a 64-node network, bandwidth demands of applications in the SPLASH benchmark suite vary from less than 0.125 Bytes/FLOP (Barnes) to 2 B/FLOP (FFT), representing a 16x difference across applications. To take advantage of this course-grain variability in the volume of traffic, a flexible network topology is needed which can allow segments[2] of on-chip routers to be turned *off* and *on* based on statically- or dynamically-determined demand for network bandwidth.

In this work, we propose such a network topology, called the Cubic Ring (*cRing*) topology. This polymorphic topology provides a simple yet flexible infra-structure for on-demand bandwidth provisioning in on-chip networks. A cRing network is obtained from a k-ary n-cube torus network by removing selected network links in all but one dimension. The resulting topology – a hierarchical arrangement of torus rings – can have different configurations, with normalized bisection bandwidth ranging from 1 bidirectional link to $k^{(n-1)}-1$. Unlike previous proposals, however, cRings can trade off network power and bandwidth without a significant increase in the average distance between nodes. This flexibility is made possible by

[1] Throughout this paper, a network resource is said to be *off* when it has been power-gated. Mechanisms proposed in [4] for powering down various network resources are assumed. A network resource is *on* if it is operating normally. This terminology is consistent with what was used in previous works, such as [5], [6] and [7].

[2] The combination of link, logic and buffers at both ends of a link is referred to as a *segment* [4].

the properties of the topology and a simple routing algorithm. *The main contribution, therefore, of this work is formalization of a new polymorphic topology which allows on-chip networks to trade off network power for network bandwidth without a significant overhead in routing complexity and zero-load latency.*

While this work does not present a complete reconfiguration-based dynamic power management scheme or the details of the microarchitecture of a polymorphic router, it motivates and presents a very suitable new topological framework (topology, routing and flow control) for such a scheme.

The rest of the paper is organized as follows: Section II surveys run-time power management schemes for on-chip networks proposed in the literature and compares the proposed cRing topology with other ring-based topologies. The cRing topology is formally described (Section III) and characterized (Section V) in its most general (n-dimensional) form. A deadlock-free routing algorithm is described and deadlock-freedom is proven in Section IV. In Section VI, performance evaluation of the cRing topology, and its comparison with other topologies that allow for on/off links, is presented. We conclude with a brief discussion on a possible router architecture suitable for the cRing topology and planned future work in Section VII.

II. Previous Work

Previous work on dynamic power management of on-chip networks has focused mostly on network links. Proposals have been made to reduce leakage power in links by turning *off* selected links in 2D mesh networks [7], reducing the width [12] of links, and applying dynamic voltage and frequency scaling (DVFS) to network links [13]. Researchers have also proposed power-gating buffers [14] and segments of on-chip routers [4]. While the work presented in [7] resembles ours most closely, polymorphic networks that use our proposed cRing topology are more suitable for exploiting coarse-grain variation in the bandwidth needs of applications. Recent studies, such as [15], have shown a greater potential for power reduction through reconfiguration at a coarse granularity (e.g., across applications) than at a fine granularity.

Several ring-based topologies have been proposed in previous work on shared memory multiprocessors. One of the most well-studied is the hierarchical slotted ring, which received significant attention several years ago in academia [16][17][18][19] as well as in industry [20][21]. Early CMP designs commonly featured the 2D mesh topology [9][10][22][23] because of its short (one-core length) network links and natural embedding in the planar 2D die layout. However, hierarchical rings have been shown to outperform 2D meshes in systems with up to 128 nodes in analytical [19][24] and experimental studies [25]. The performance gain is realized primarily because of two factors: the memory access locality of the workloads, and the simplicity of ring routers compared to mesh routers. The proposed cRing network is different from hierarchical rings in that it allows for more than one ring to connect two levels of the hierarchy. As such, the cRing topology avails both the locality-friendliness and simplicity of the hierarchical ring topology as well as the increased bisection bandwidth of the network through additional rings to connect each level of the hierarchy.

III. Formal Description

A. Cubic Ring Networks

The cubic ring topology can best be described formally by comparing it with the more well-known k-ary, n-cube torus topology. A cRing network is obtained from a torus network by removing selected communication channels in all but one dimension. Stated formally, a torus network is a graph $I_T(N_T, C_T)$, where vertices of the graph N_T represent the nodes (routers) in the network, and the edges of the graph C_T represent the communication channels (links) connecting the nodes. A cRing network $I_C(N_C, C_C)$, derived from I_T, has N_C = N_T and $C_C \subset C_T$.

A bidirectional n-dimensional, radix-k torus, or k-ary n-cube torus, consists of $N = k^n$ nodes arranged in an n-dimensional lattice with k nodes along each dimension. Each node, A, has a unique n-digit, radix-k address $(a_{n-1}, a_{n-2}, ..., a_1, a_0)$ that denotes its location in the lattice [26]. Each node is connected via bidirectional channels to $2n$ nodes with addresses that differ by ±1 (mod k) in exactly one address digit. That is, any two nodes, A and B, are connected to each other *iff* $b_i = a_i$ for all $0 \leq i \leq n-1$ except one, j, where $b_j = a_j \pm 1$. This results in a total of nk^n bidirectional inter-node channels in the network, organized as nk^{n-1} bidirectional *rings* each of size k. For example, consider the networks in the $4 \times 4 \times 4$ lattice shown in Figure 1. We use Cartesian coordinates to simplify the graphical representation of the networks and their explanation. Figure 1a shows 64 nodes connected in a 4-ary 3-cube torus topology.

An n-dimensional, radix-k, R-ring cubic ring network, or n-k-R cRing for short, can be derived from the corresponding n-dimensional, radix-k torus by removing a subset of rings from the torus topology in a hierarchical fashion. If a ring connecting a given node to its neighbors in the i-th dimension is removed, all rings incident on the node in dimensions $i' > i$ are also removed. This property has the effect of creating a hierarchy of rings where each dimension becomes a *level* in the hierarchy. In this hierarchy, all the rings in a given dimension are at the same hierarchical level and each ring in a given dimension has at least one node that is part of another ring in the next higher dimension, if there exists one. The network is fully connected if and only if at least one ring in each dimension is connected and all the rings in the lowest dimension, *dimension 0*, are connected. Dimension 0 does not imply a particular dimension in the Cartisan coordinates. Any dimension in which all rings are connected can become the lowest hierarchical level, or dimension 0. We discuss the ordering of dimensions in more detail in Section III-C.

In the notation for cRing networks, $R = \{r_{n-1}, r_{n-2}, ..., r_1, r_0\}$, where r_i, $0 \leq i \leq n-1$, is a k-bit string. The l-th bit of each string r_i, $0 \leq l \leq k-1$ (bit 0

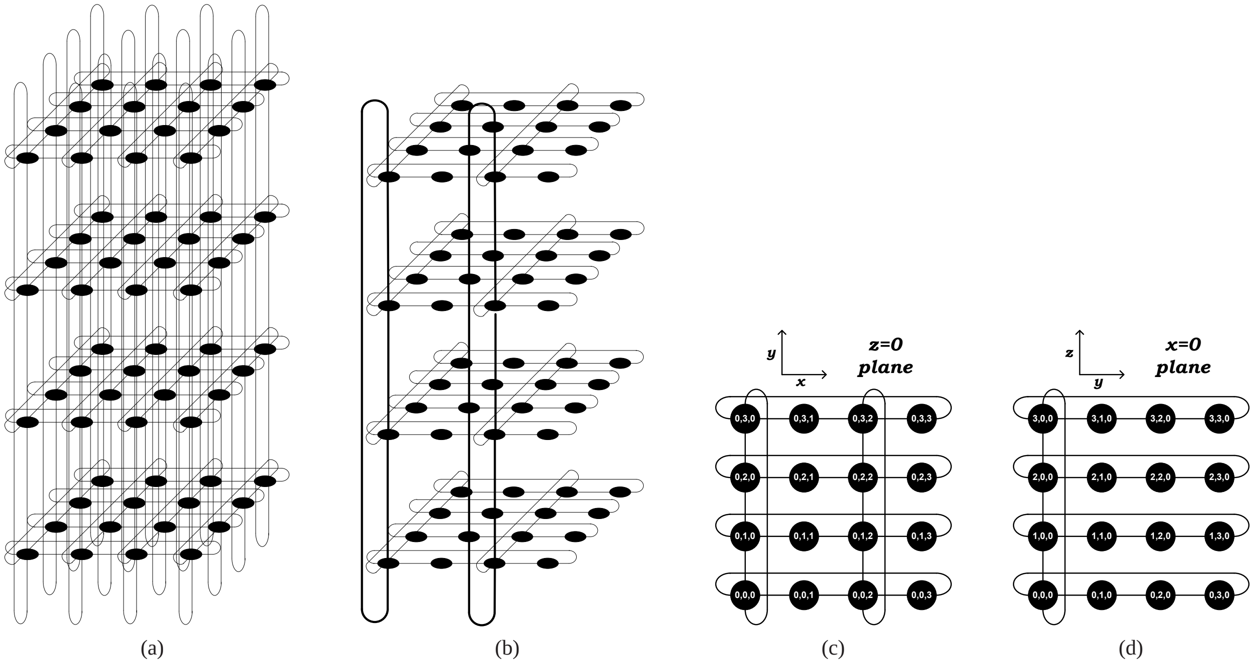


Fig. 1. A 64-node (a) 4-*ary*, 3-*cube* torus network, (b) 4-*ary*, 3-*cube*,*R*-*ring* cubic ring network with $R = \{0001, 0101, 1111\}$, (c) the xy plane of the 4-ary, 3-cube cRing network at $z = 0$, and (d) the yz plane at $x = 0$.

being the LSB and bit $k-1$ being the MSB) corresponds to a specific set of one or more torus rings in the i-th dimension, and the *value* of each bit indicates the presence (if *value* = 1) or absence (if *value* = 0) of the corresponding set of rings (or ring, if $i = n$) in the cRing topology. That is, if $r_i[l]$ = 0, the set of rings connecting nodes with $a_{i-1} = l$ in the i-th dimension is removed from the corresponding torus. This also means that nodes with $a_{i-1} = l$ do not have any active channel in the dimensions greater than i. We have that $r_0[l] = 1$ for all l, $0 \leq l \leq k-1$, so the expression $a_{i-1} = l$, which is meaningless for $i = 0$, applies only for $i > 0$. Finally, for all i, $0 < i \leq n-1$, $r_i[l] = 1$ for at least one value of l.

Consider the example of the cRing network shown in Figure 1b. This 64-node, 4-3-R cRing network with $R = \{0001, 0101, 1111\}$ shows how higher-dimensional cRing networks can be constructed. In this network, each x-dimension ring connects to two rings in the y dimension; at $x = 0$ and $x = 2$ as indicated by $r_1[0] = r_1[2] = 1$. This results in eight y-dimension rings connecting the sixteen x-dimension rings across the four xy planes. Figure 1c shows the xy plane at $z = 0$ as an example. Finally, each y-dimension ring connects to one ring in the z dimension, as indicated by $r_2[0] = 1$. This results in the two z-dimension rings connecting the eight y-dimension rings, across the two yz planes at $x = 0$ and $x = 2$. Figure 1d shows the yz plane at $x = 0$. The plane at $x = 2$ would be identical, whereas the planes at $x = 1$ and $x = 3$ will not have any nodes connected in the y or z dimension.

The network shown in Figure 1b has three levels of hierarchy corresponding to the three dimensions. Rings in the x dimension constitute the lowest level of the hierarchy, rings in the y dimension constitute the next higher, and finally the rings in the z dimension constitute the highest level of the hierarchy. In the rest of the paper, we refer to ring(s) in the lowest level (x-dimension in Figure 1b) as “local” ring(s), ring(s) at the next higher level (y-dimension in Figure 1b) as “intermediate” ring(s), and ring(s) in the highest level (z-dimension in Figure 1b) as “global” ring(s). A cRing with $n > 3$ will have multiple “intermediate” levels, denoted as *intermediate1* for ring(s) at $i = 1$, *intermediate2* for ring(s) at $i = 2$, and so on. Network nodes are classified based on the highest level ring of which they are a part. So, in the example network shown in Figure 1b, node (0,0,0) is a “global ring node” as the highest level ring of which it is a part is the global ring. Similarly, node (0,1,0) is an “intermediate ring node” and node (0,0,1) is a “local ring node.” Finally, the local ring on which the source (destination) of a packet lies is called the “source (destination) local ring” and packets for which the source and the destination lies on the same local ring are referred to as “local packets.”

B. *Mixed-Radix and Isomorphic Cubic Ring Networks*

Like mixed-radix torus networks, each dimension in a cubic ring network may have a different radix. This will mean a different k for each dimension. Figure 2a shows a 2,4-ary 2-cube R-ring cRing with $R = \{0101, 11\}$. An isomorphic pair of this mixed-radix network is shown in Figure 2b. Two cRing networks are isomorphic if the r_i, $0 \leq i \leq n-1$, of one network can be obtained by rotating the r_i of the other. The network shown in Figure 2c is not isomorphic to the other two networks. Isomorphic cRing networks have identical performance characteristics like average and maximum hop count between two nodes and maximum throughput.

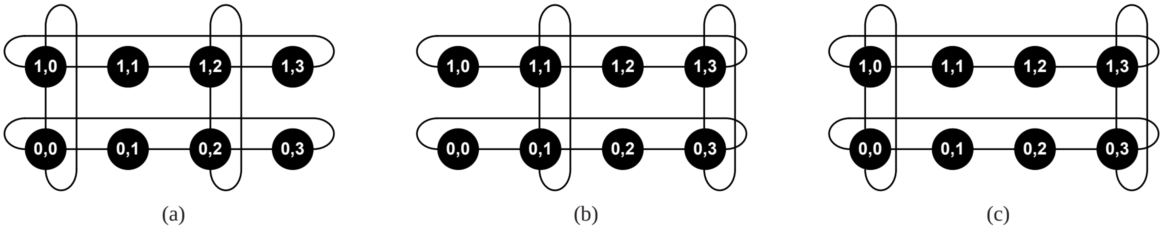


Fig. 2. An 8-node, 2,4-ary 2-cube R-ring cRing network with (a) $R = \{1010, 11\}$, (b) $R = \{0101, 11\}$, and (c) $R = \{1001, 11\}$.

C. *Polymorphic Cubic Ring Network*

The polymorphic capability of cubic ring networks enables intermediate and global rings to be turned *on* and *off* as network bandwidth demand varies. For example, when the bandwidth demand is high, all intermediate and global rings can be are turned *on* to allow the network to operate in a fully-configured torus mode. When bandwidth demand is low, e.g., determined statically (offline) or dynamically (online), selected intermediate and global rings can be turned *off* to allow the network to operate in one of the sparsely-connected cRing modes. A description of the techniques that can be used to determine when and how the polymorphic cRing network should be configured–whether static (offline) or dynamic (online) techniques–is beyond the scope of this paper. However, regardless of which configuration the cubic ring network should take on, the same routing algorithm described below can be used to transport packets across the network from any source to any destination.

IV. Routing Function for Cubic Ring Networks

Hierarchical arrangement of rings in the cRing topology simplifies design of the routing algorithm. The route that each message takes from its source to its destination can be decomposed into (at maximum) two segments – the *up* segment and the *down* segment. In the *up* segment of the route, the message tries to reach the highest dimension in which the source and the destination differ. This is done by routing to the nearest node that connects in the next higher dimension, and so on. Once the highest dimension in which the source and the destination nodes differ is reached, the message travels down the hierarchy of rings, crossing dimensions in strictly decreasing order, reducing to zero the offset in one dimension before routing in the next until the destination node is reached. In effect, the route in the *down* segment is identical to that supplied by the *e-cube* routing function [27]. If the message is injected into the network at the node connected in the highest dimension in which the source and destination differ, there is no need for the message to travel up the hierarchy. Therefore, its route to the destination will consist only of the *down* segment.

The cRing routing function described above is connected because each *ring* at every level of the hierarchy is guaranteed to have at least one node with a link to the next higher level (if there exists one). This means that a message can traverse dimensions in increasing order starting from the lowest dimension (i.e., the dimension in which all rings are connected) until the desired dimension is reached. Conversely, each node at a given level of the hierarchy is part of exactly one ring at the next lower level (if there exists one). This means that a message can follow the minimal path as it travels down the hierarchy, traversing dimensions in decreasing order until the destination node is reached.

A. *Preliminaries*

Before a formal description of the cRing routing function and the proof of deadlock freedom are presented, some notation is provided below.

- Let m be a message with source $s = (s_{n-1}, s_{n-2}, ..., s_2, s_1, s_0)$ and destination $d = (d_{n-1}, d_{n-2}, ..., d_2, d_1, d_0)$.
- Let $\Delta = \{\Delta_{n-1}, \Delta_{n-2}, ..., \Delta_1, \Delta_0\}$ be the offset between the source s and destination d of m such that $\Delta_i = (d_i - s_i) \ mod \ k$, for all $0 \leq i \leq n-1$.
- Let h be the highest dimension in which the source and the destination differ. That is, $\Delta_h \neq 0$ and either $h = n-1$ or $\Delta_{h'} = 0$, for all $h < h' \leq n-1$.
- Let $u = (u_{n-1}, u_{n-2}, ..., u_1, u_0)$ be the current node where m is queued. The offset between the current node and the destination is $\delta = \{\delta_{n-1}, \delta_{n-2}, ..., \delta_1, \delta_0\}$, where $\delta_i = (d_i - u_i) \ mod \ k$, $0 \leq i \leq n-1$. For $u = s$, $\delta = \Delta$.
- Let the highest dimension in which the current node u is connected be χ.

B. *cRing Routing Function*

The cRing routing algorithm is shown in Figure 3. Messages are created with their direction set to *up*. The Route-Message(m) procedure first determines the correct direction of the message (using the Get-Direction(m) procedure). If the direction is *down*, the message is routed using *e-cube* routing, traversing dimensions in decreasing order such that δ_i is reduced to zero in each dimension i. If the direction is *up*, the message is routed toward the nearest node that connects in a higher dimension. At each node in the network, the output channel that takes a message toward the node that connects in a higher dimension is fixed and independent of the message destination. Therefore, up messages do not undergo routing like the down messages do; the output channel for *all* up messages is known *a priori*.

To avoid routing-induced deadlocks *across* rings, two virtual channels, VC0 and VC1, are used in the network. VC0 is

```
Procedure Route-Message (m)

0.  if (u == d), consume message

1.  π = Get-Direction(m)

2.  if (π = down), route m using the
    e-cube hop

3.  if (π = up), route m along dimension
χ toward a node with χ ', where χ ' > χ
```

```
Procedure Get-Direction (m)

if (direction == down)
    return (down);

elseif (direction == up)
    if (χ < h)
        return (up);
    else
        return (down);
```

Fig. 3. The cRing routing algorithm.

reserved for messages travelling in the *up* direction and VC1 is reserved for messages travelling in the *down* direction. Bubble flow control [28] is used on both virtual channels to avoid deadlocks *within* rings. This means that new messages being injected into the network, as well as messages making a turn from one dimension to the other, must satisfy the bubble condition before they can make progress (more details on the use of bubble flow control appear in the next section). Starvation is prevented through the use of the mechanism proposed in [28].

Example: As an example, consider the routing of a message from source s = (0,1,1) to destination d = (2,3,2) in the 4-3-R cRing network, with $R = \{0001, 0001, 1111\}$, shown in Figure 4. The offset between the source and the destination for this message is $\Delta = \{2, 2, 1\}$, and h = 2.

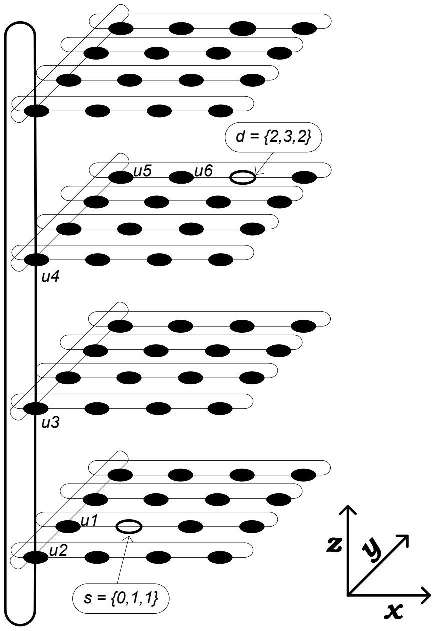


Fig. 4. Routing of a message in a 4-ary 3-$cube$ cRing with $R = \{0001, 0001, 1111\}$.

The path taken by the message is: $s = (0, 1, 1) \rightarrow u_1 = (0, 1, 0) \rightarrow u_2 = (0, 0, 0) \rightarrow u_3 = (1, 0, 0) \rightarrow u_4 = (2, 0, 0) \rightarrow u_5 = (2, 3, 0) \rightarrow u_6 = (2, 3, 1) \rightarrow d = (2, 3, 2)$.

At the source s, $\chi = 0$ because the node (0,1,1) is connected only in the 0^{th}-dimension (i.e., the x-dimension). This means that $\chi < h$, resulting in the message direction to be set as *up*. This causes the message to be routed toward u_1, the nearest node that connects to a higher dimension. At u_1, $\chi = 1$ but χ is still less than h, so the message direction remains *up*. At u_1, the nearest node with a higher χ value is u_2 so the message is routed toward u_2. At u_2, $\chi = h = 2$, so, the message direction is changed to *down*. All subsequent hops are governed by the *e-cube* routing algorithm. The δ at u_2 is $\{$2,3,2$\}$, so the message first makes two hops in the z+ direction (to u_3 and u_4), then at u_4, the message makes a hop in the y- direction (to reach u_5), and finally at $u5$ the message makes two hops in the x+ direction to reach the destination d.

From s to u_1 and from u_1 to u_2, the message travels on VC0. At u_2, the message switches to VC1 and remains on VC1 until it is consumed at the destination.

It is important to note that if the cRing routing algorithm were to be used to route messages on a torus topology, the effective routing function would be e-cube, since $h = \chi$ at each (injection) node.

C. Deadlock Freedom

We now prove that cRing routing as described above is deadlock-free using only two virtual channels and bubble flow control.

Lemma 1: Ring cycles in each dimension are broken by using bubble flow control.

Proof: Bubble flow control works on the principle that a message can be injected into a ring *iff* after the injection of the message there remains at least one empty message buffer (or bubble) in the ring. It has been proven that bubble flow control is sufficient to avoid *within ring* deadlocks in torus networks [29].

Lemma 2: The cRing routing algorithm is deadlock-free for messages traveling in the *up* direction.

Proof: For deadlock to occur, there has to be a cyclic dependency on virtual channels acquired by the messages involved in the deadlock. Messages going in the *up* direction use VC0 and can request resources either in the current dimension or a higher dimension. This results in a total ordering of resources that messages in VC0 can request. In the routing example given in Figure 4, this ordering is x+/x- → y+/y- → z+/z-. This total ordering leads to a channel dependency graph which is acyclic except for the ring cycles, and we have shown in the proof of Lemma 1 that those cycles are broken by bubble flow control.

Lemma 3: The cRing routing algorithm is deadlock-free for messages traveling in the *down* direction.

Proof: A corollary to the statement of Lemma 2 is that messages going in the *down* direction, using VC1, also cannot cause a deadlock to occur as there is a total order in which resources can be requested by the *down* messages. Again using the example given in Figure 4, the order in which queues can be requested by messages going in the down direction is z+/z- → y+/y- → x+/x-. This results in an acyclic dependency graph with the same exception as stated in the proof of Lemma 2. Therefore, the reasoning of Lemma 2 holds for Lemma 3.

Theorem: The cRing routing algorithm is deadlock-free.

Proof: Routing of *up* messages (VC0) is deadlock-free (Lemma 2) and routing of *down* messages (VC1) is deadlock-free (Lemma 3). Hence, the only way the cRing routing function could be deadlock-susceptible is if there can be a cyclic dependency *across* messages in VC0 and VC1. *Up* messages can request resources reserved for the *down* messages when they switch their direction (i.e., there is a direct dependency from VC0 to VC1) but *down* messages always sink and cannot request resources reserved for *up* messages (i.e., there is no dependency from VC1 to VC0). As dependencies between VC0 and VC1 resources are acyclic, the cRing routing function is deadlock-free.

V. Characterization of cRing Networks

Unlike k-ary n-cube tori, cRing networks are not regular (i.e., all nodes do not have the same degree), and they possess neither edge nor node symmetry. This lack of regularity and symmetry makes the derivation of generalized expressions for the computation of analytical performance metrics like average and maximum distance, network capacity and maximum channel load very difficult. Below, we present approximate measures of average and maximum latency and bisection bandwidth.

A. Average Distance

In formulating the average distance (or average hop count) for a cRing network, we exploit the hierarchical arrangement of the network and the *up* and the *down* segments of a packet's path. To facilitate derivation, we make two simplifying assumptions which are later removed. First, we assume that k is even. Second, it is assumed that the number of rings connecting dimension i to dimension $i+1$ is one for all $0 < i < n-1$. In terms of the notation described in Section III-A, this means that the number of 1's in each string $r[i]$ is exactly one.

The average distance between two nodes in a 2D cRing network (γ_{2D}) can be approximated by the following expression.

$$\gamma_{2D} = \frac{2k}{4} + \frac{2k}{4}(1 - \frac{k}{k^2}) \qquad (1)$$

In a torus network, the average distance between any two nodes along the shortest path is given by $\frac{nk}{4}$. In cRing networks, once a packet reaches the highest level in the hierarchy in which its source and destination nodes differ, it can take the shortest path downwards. This means that the average distance for the downward segment of each packet's path is equal to $\frac{nk}{4}$. However, the upward segment of a packet adds additional distance to the total average distance in a cRing network. In Equation 1, we add this additional distance by taking the product of the average distance from any given node to the global ring node in each local ring ($\frac{2k}{4}$) and the fraction of traffic that must exit the local ring ($1-\frac{k}{k^2}$), where k is the number of nodes in each ring and k^2 is the total number of nodes in a 2D cRing network.

Generalizing Equation 1 to n dimensions gives us the following expression for average distance.

$$\gamma_{nD} = \frac{nk}{4} + \sum_{i=1}^{n-1} \frac{k}{4}(1 - \frac{k^i}{k^n}) \qquad (2)$$

Equation 2 considers cRing networks with only one ring in each dimension i, where $i > 0$. To generalize this expression further, cRing networks with more than one ring in each dimension (above dimension 0) must also be considered. This complicates the analysis, and therefore, we approximate this by the following expression.

$$\gamma_{nD} \simeq \frac{nk}{4} + \sum_{i=1}^{n-1} \frac{k}{2 \cdot 2^{numRi}}(1 - \frac{k^i}{k^n}) \qquad (3)$$

In Equation 3, the number of rings in dimension i are given by $numRi$, and the expression assumes that $numRi$ rings are maximally apart.

Finally, in the above expressions, $\frac{k}{4}$ can be substituted by $\frac{k}{4} - \frac{1}{4k}$ if k is odd.

B. Maximum Distance

Similarly, the maximum distance, or diameter, between two nodes can be approximated by the following expression.

$$\gamma_{max} \simeq \frac{nk}{2} + \sum_{i=1}^{n-1} \frac{k}{2^{numRi}}(1 - \frac{k^i}{k^n}) \qquad (4)$$

C. Bisection Bandwidth

The hierarchical nature of cRing networks implies that, regardless of the configuration, the least connected cut of the graph representing the network will always be along the highest dimension. Therefore, the bisection bandwidth of a cRing network is equal to the connected rings in the n-th dimension, or $numRn$ from the definition above.

VI. Evaluation

We evaluate the performance of cRing networks in various configurations and compare their performance to *on*/*off* networks proposed in the literature. But first, we quantify the power-bandwidth trade-off offered by cRing networks. To do so, we investigate the effect of adding/removing global rings on average distance of a cRing network. Given that this work is focused on on-chip networks, we limit our analysis to 2D cRing networks, but the analysis can easily be extended to higher dimensional networks.

Figure 5 plots the average hop count of 16- and 64-node cRing networks in all possible (optimal) configurations. An optimal configuration is one in which the global rings are spaced maximally apart to reduce average distance. The plot shows an important feature of cRing networks: there is a significant decrease in average distance when the number of global rings is increased from 1 to 2. However, as the number of global rings is increased further, there is a diminishing effect on average distance. For example, the 64-node cRing network with only four global rings has only 2.75% higher

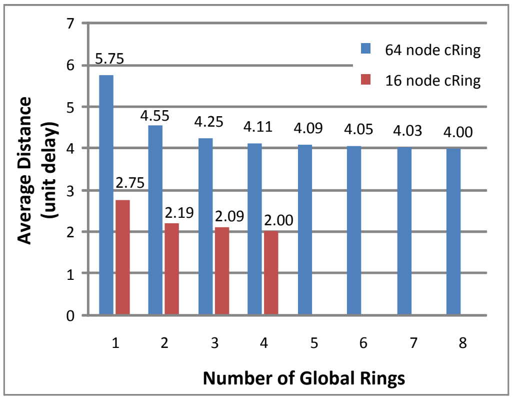


Fig. 5. Average distance for 16- and 64-node cRing networks in all configurations, with optimally placed global rings.

average distance than the 64-node torus. This trend is less pronounced in smaller networks (e.g., the 16-node network also characterized in the plot) but becomes further pronounced in larger networks, as shown in Figure 6.

Figure 6 shows an important trend that fundamentally motivates cRing topologies. The number of global rings necessary to keep the average distance within 5% of the average distance of a torus network is insensitive to the size of the network. With only four of the 16 available global rings connected, a 16×16 (256-node) cRing network has an average distance of 8.3, which is only 3.75% higher than the average distance of the 16-*ary*, 2-*cube* torus (average distance = 8). With only 6 global rings (i.e., with 31% of the links turned *off*), the increase in average distance drops to a mere 1.65%. This shows that additional global rings contribute mainly to increasing network bandwidth, rather than to reducing average distance. Therefore, by turning *off* these global rings when bandwidth demands of the network permit it, significant reduction in network (static) power consumption is achieved.

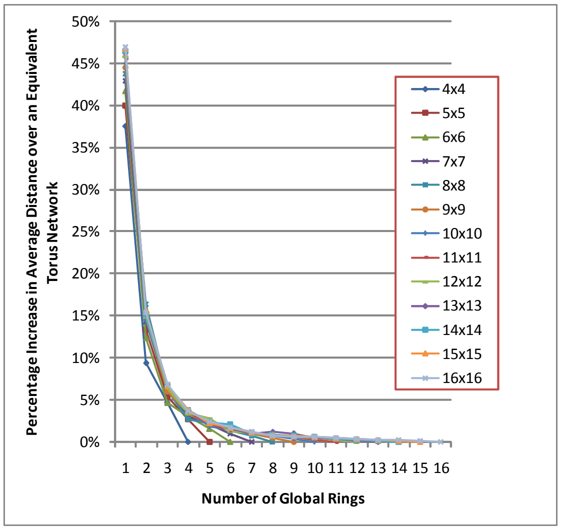


Fig. 6. Percentage increase in average distance from the torus network in cRing networks of different sizes and different number of global rings.

Figure 7 illustrates this more clearly by plotting the percentage of network links that are turned *off*, as a fraction of the total number of links in the 16-ary, 2-$cube$ torus, along with the percentage increase in average distance from the equivalent torus.

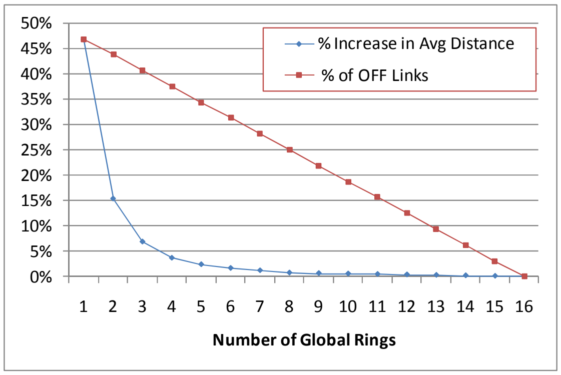


Fig. 7. Percentage of network links turned *off* and average distance for 256-node cRing networks in all configurations.

Next, we quantify the savings in network power that can be realized from turning *off* rings in a cRing topology. Turning *off* global rings lowers static power in two ways: turned *off* links save leakage power in links, and input and output ports associated with these turned *off* links can also be power-gated to reduce static power in routers. Link power savings is simply proportional to the percentage of links turned *off*, so little further investigation is necessary.

To quantify static power savings in routers, we implemented a polymorphic router in Verilog HDL and synthesized it using Synopsys Design Compiler, targeting TSMC 90nm technology. The router has 2 virtual channels, flit size of 64-bit and 8-flit input buffers at each port. In the 3-port configuration, input and output ports associated with the *off* global rings are power-gated. Table I shows router power consumption in 5- and 3-port configurations.

Parameter	**5-Port Config**	**3-Port Config**	**Diff**
Cell Area (μm^2)	183,000	105,344	-42.4%
Static Power (mW)	54.39	33.37	-38.6%

TABLE I

COMPARISON OF POLYMORPHIC ROUTER ACTIVE CELL AREA AND STATIC POWER IN 5-PORT AND 3-PORT CONFIGURATIONS.

Aggregating the per-router power over the entire network highlights the power-performance trade-off afforded by cRing networks. A 4-ary, 2-$cube$ cRing network with two global rings has 19.3% lower static power consumption than the equivalent torus network. This saving, however, comes at the expense of about 9% higher average distance. Increased average distance indicates not only higher average latency but, more importantly in this case, increased dynamic power. As the size of the network increases, however, the trade-off looks very different and more favorable for the cRing network. For example, for an 8-ary, 2-$cube$ cRing network, a 19.3% reduction in static power can be achieved with 3 global rings and only a modest 2.75% increase in the average distance.

Further savings in router power, especially in dynamic power, are also possible through careful design of a partitionable router crossbar and switch allocator. In future work, we will evaluate the microarchitectural design choices for a polymorphic router, as well as the power overhead of powering *on* and *off* global rings.

Finally, throughout this section we have assumed cRing networks with optimally-placed global rings. Optimal placement, as mentioned earlier, implies that intermediate and global rings are placed maximally apart. We studied the impact of sub-optimal placement of global rings on the average distance in cRing networks and found that in a 64-node cRing network, worst-case placement of global rings increases the average distance by up to 17%. This points to the importance of spacing global rings maximally apart.

A. Performance Comparison

We compare the performance of cRing networks with *on*/*off* networks based on the West-Last, East-Last (WLEL) routing algorithm proposed in [7]. The simulation environment used in this study is based on a flit-level network simulator SICOSYS (SImulator of COmmunication SYStems)[30]. SICOSYS is a detailed network simulator written in C++ that incorporates key parameters of the low-level implementation and provides results close to those from Verilog/VHDL simulators, but at lower computational cost. In a previous study, it has been shown that, compared to an RTL description of a router, results from SICOSYS simulations had less than 4% error in latency and even less in throughput. This accuracy came with up to 45× speed-up over RTL simulation.

Simulations presented here use the basic architecture of a 4-stage Bubble Adaptive Router [31] with link latency of one cycle. Two flit packets were assumed and simulations were run for 100,000 cycles with a warm-up period of 10,000 cycles. Three synthetic traffic patterns were used in these experiments: uniform random traffic, perfect shuffle traffic, and local (near neighbor) traffic where packets are sent only to the nodes due east, west, north and south of the injecting node. Injected load rate was normalized to the bisection bandwidth of a torus network with values ranging from [0,1]. A network was assumed to be saturated when the average packet latency exceeded 2× the zero-load latency.

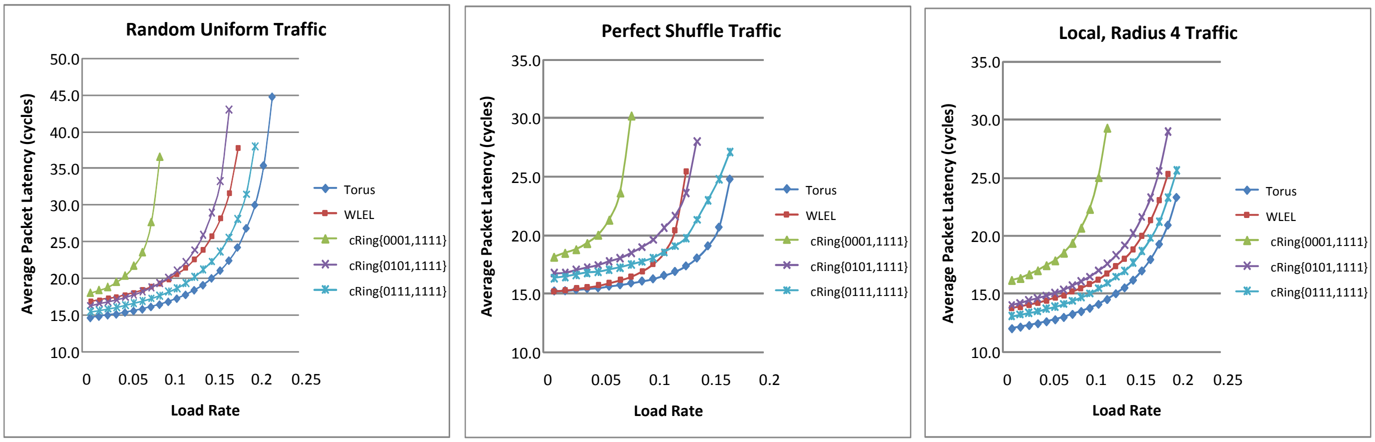
Figure 8 shows the performance of the following five 16-node networks: (a) A 2D torus network, (b) 4-*ary*, 2-*cube* cRing network with R=$\{0001,1111\}$, (c) with R=$\{0101,1111\}$, (d) with R=$\{0111,1111\}$, and finally (e) a 16-node 2D torus network that uses West-Last, East-Last routing algorithm proposed in [7] with the maximum number of links turned *off*. The average packet latency of a cRing network with two global rings is about 11% higher than that of a torus under uniform random traffic, 10% higher under perfect shuffle traffic and 16.7% higher under local traffic patterns. Given that only 12.5% of the router segments are turned *off* in this configuration, the power-performance (latency) trade-off is linear. The results indicate that cRing networks are not well-suited for smaller networks, as the latency penalty of turning *off* rings is quite high.

Figure 9 shows results for a network size of 64 nodes. Three cRing configurations were simulated with one, two and three global rings, respectively. With only three global rings active (i.e., almost 30% of the router segments turned off) the latency overhead is only 10.6% with uniform random traffic, 11.3% with perfect shuffle traffic and 16.5% with local traffic patterns. What is important to note here is that the simulator assumes the same router delay in 5-port and 3-port mode. A dimensionally-partitioned router can be designed to have lower router delay in 3-port mode than in 5-port mode. While the focus of this paper is not on the design of a partitionable router, the results presented here show that even a modest reduction in router delay in the 3-port configuration can make the latency overhead almost negligible for a 64-node network even with just two or three global rings. The torus network which uses WLEL routing performs well under perfect shuffle traffic because this traffic pattern is well-matched to the set of *on* links. For all other traffic patterns, however, the torus network using WLEL performs worse than the cRing network with three global rings.

Finally, Figure 9 also shows the throughput penalty of cRing networks. Under uniform random traffic, for example, while the latency penalty for a cRing network with three global rings is lower than that for a torus with WLEL routing, the cRing network has lower saturation throughput. This, however, is not problematic because as the demand for network bandwidth increases, the polymorphic cRing network will turn *on* additional global rings, thus increasing the maximum throughput, until all global rings have been turned *on* and the network is restored to a fully-connected torus topology. In other words, a polymorphic network only operates in the degraded cRing mode when the the throughput needs of the application are well below the throughput available in the torus configuration.

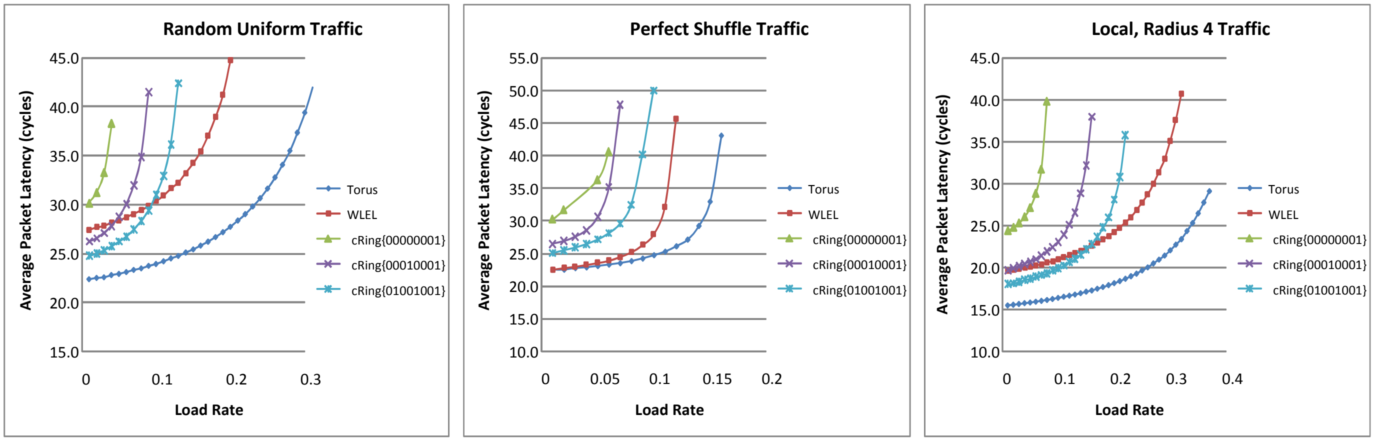
VII. Discussion and Conclusion

An *ideal* on-chip network whould consume power only when it is delivering packets. However, in practice, such efficiency is difficult to achieve primarily because of the delay and energy overhead in turning network resources *on* and *off*. Practical on-chip networks have, therefore, used only fine-grain dynamic power management techniques that can conserve power only during short periods of inactivity on a link-by-link basis. But these techniques are not suited to take advantage of significant variations in bandwidth needs across applications. Coarse-grain power management approaches, such as the one enabled by the polymorphic cRing networks described in this paper, allow for greater savings as a significant number of network resources can be turned *off* for prolonged periods of time. However, in order to be effective, these techniques must



(a) (b) (c)

Fig. 8. Performance results comparing a 16-node two-dimensional torus network, a cRing network, and a torus network that uses West-Last, East-Last routing, under (a) random uniform, (b) perfect shuffle and (c) local traffic with radius 4.



(a) (b) (c)

Fig. 9. Performance results comparing a 64-node two-dimensional torus network, a cRing network and a torus network that uses West-Last, East-Last routing, under (a) random uniform, (b) perfect shuffle and (c) local traffic with radius 4.

maintain low latency even when the network is in the low-power state(s), and they must provide high throughput during normal operation.

This work lays the foundation for such an effective coarse-grain power management approach by presenting a flexible polymorphic topology which can be used to trade off power for bandwidth without significantly increasing average packet latency. The definition of the topology presented in this work is general enough to accommodate 3D-die stacks and heterogenous core sizes. The routing algorithm is also formalized, and proven to be correct and deadlock-free for the general case. Also in this work, through analytical and experimental performance evaluation, key advantages of the proposed cRing topology are shown, and important cost-performance trade-offs are highlighted.

In future work, we will implement a methodology to dynamically *morph* between a torus network and various cRing configurations. The router architecture and network management mechanism necessary to do so will be explored, and the power-savings quantified. Finally, we plan to explore the performance of various cRing configurations under real application workloads.

The work presented in this paper also has application beyond the power management scheme that constitutes our future work. The cRing routing function can, for example, be used to design more fault tolerant torus networks. Such networks will be able to tolerate multiple permanent faults in links and associated buffers as long as at least one of the cRing configurations can be realized from the connected links. Given the small fraction of torus links that are necessary to realize a cRing network, such networks can have good fault coverage at minimal cost. cRing networks can also be viable candidates for application-specific SoCs, where traffic locality can be mapped to topological locality. The flexibility of local rings of unequal sizes and multiple global rings – coupled with familiar 3- and 5-port router architectures – makes polymorphic cRing networks important additions to the toolbox of chip designers.

Acknowledgment

The research described in this paper was supported, in part, by National Science Foundation (NSF) grants CCF-0541417 and CCF-0836759.

References

[1] D. Burger and J. R. Goodman. Billion-Transistor Architectures: There and Back Again. *IEEE Computer*, 37(3):22–28, March 2004.

[2] Tilera Announces the World's First 100-core Processor (TILE-Gx100). *www.tilera.com/news_&_events/press_release_091026.php*.

[3] S. Borkar. Thousand Core Chips: A Technology Perspective. In *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE*, pages 746–749, June 2007.

[4] B. Grot Kyle Hale and Steve Keckler. Segment Gating for Static Energy Reduction in Networks-On-Chip. In *Proceedings of the 2nd International Workshop on Network on Chip Architectures (NoCArc)*, 2009.

[5] Vassos Soteriou and Li-Shiuan Peh. Dynamic Power Management for Power Optimization of Interconnection Networks Using On/Off Links. In *Proceedings of the 11th Symposium on High Performance Interconnects*, August 2003.

[6] Vassos Soteriou and Li-Shiuan Peh. Design-Space Exploration of Power-Aware On/Off Interconnection Networks. In *Proceedings of the 22nd International Conference on Computer Design (ICCD)*, October 2004.

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

[8] Jason Sungtae Kim, Michael Bedford Taylor, Jason Miller, and David Wentzlaff. Energy Characterization of a Tiled Architecture Processor with On-chip Networks. In *Proceedings of the 2003 International Symposium on Low-Power Electronics and Design (ISLPED'03)*, pages 424–427, New York, NY, USA, 2003. ACM.

[9] Michael Bedford Taylor, Walter Lee, Saman Amarasinghe, and Anant Agarwal. Scalar Operand Networks: On-Chip Interconnect for ILP in Partitioned Architectures. In *Proceedings of the The Ninth International Symposium on High-Performance Computer Architecture (HPCA'03)*, page 341. IEEE Computer Society, 2003.

[10] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote and N. Borkar. An 80-Tile 1.28TFLOPS Network-on-Chip in 65nm CMOS. In *IEEE International Solid-State Circuits Conference. Digest of Technical Papers.*, Feb. 2007.

[11] S. C. Woo, M. Ohara, E. J. Torrie, J-P. Singh, and A. Gupta. The SPLASH-2 Programs: Characterization and Methodology Considerations. In *Proceedings of the 22nd International Symposium on Computer Architecture*, pages 24–36. IEEE Computer Society Press, June 1995.

[12] Marina Alonso, Juan Miguel Martinez, Vicente Santonja and Pedro Lopez. Reducing Power Consumption in Interconnection Networks by Dynamically Adjusting Link Width. In *Proceedings of 2004 EuroPar Conference*. Springer-Verlag, 2004.

[13] Dongkun Shin and Jihong Kim. Power-Aware Communication Optimization for Networks-On-Chips with Voltage Scalable Links. In *Proceedings of the 2nd IEEE/ACM/IFIP International Conference on Hardware/Software Co-Design and System Synthesis*, pages 170–175. ACM, 2004.

[14] Xuning Chen and Li-Shiuan Peh. Leakage Power Modeling and Optimization in Interconnection Networks. In *ISLPED '03: Proceedings of the 2003 international symposium on Low power electronics and design*, pages 90–95, New York, NY, USA, 2003. ACM.

[15] Héctor Montaner, Federico Silla, Vicente Santonja, and José Duato. Network Reconfiguration Suitability for Scientific Applications. In *ICPP '08: Proceedings of the 2008 37th International Conference on Parallel Processing*, pages 312–319, Washington, DC, USA, 2008. IEEE Computer Society.

[16] M. Holliday and M. Stumm. Performance Evaluation of Hierarchical Ring-Based Shared Memory Multiprocessors. *IEEE Transaction on Computers*, 43(1):52–67, 1994.

[17] Xiaodong Zhang and Yong Yan. Comparative Modeling and Evaluation of CC-NUMA and COMA on Hierarchical Ring Architectures. *IEEE Transaction on Parallel and Distributed Systems*, 6(12):1316–1331, December 1995.

[18] Keith Farkas, Zvonko Vranesic, and Michael Stumm. Scalable Cache Consistency for Hierarchically Structured Multiprocessors. *Supercomputing*, 8(4):345–369, 1995.

[19] V. Carl Hamacher and Hong Jiang. Hierarchical Ring Network Configuration and Performance Modeling. *IEEE Transaction on Computers*, 50(1):1–12, 2001.

[20] D.R. Cheriton, H.A. Goosen, and P.D. Boyle. Paradigm: A Highly Scalable Shared-Memory Multicomputer Architecture. *Computer*, 24(2):33–46, Feb 1991.

[21] A. W. Wilson. Hierarchical Cache/Bus Architecture for Shared Memory Multiprocessors. In *Proceedings of the 14th Annual International Symposium on Computer Architecture*, pages 244–252, 1987.

[22] Karthikeyan Sankaralingam, Ramadass Nagarajan, Robert McDonald, Rajagopalan Desikan, Saurabh Drolia, M.S. Govindan, Paul Gratz, Divya Gulati, Heather Hanson, Changkyu Kim, Haiming Liu, Nitya Ranganathan, Simha Sethumadhavan, Sadia Sharif, Premkishore Shivakumar, Stephen W. Keckler, and Doug Burger. Distributed Microarchitectural Protocols in the TRIPS Prototype Processor. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 39)*, December 2006.

[23] D. Wentzlaff, P. Griffin, H. Hoffmann, Liewei Bao, B. Edwards, C. Ramey, M. Mattina, Chyi-Chang Miao, J. F. Brown, and A. Agarwal. On-Chip Interconnection Architecture of the Tile Processor. *Micro, IEEE*, 27(5):15–31, September 2007.

[24] V. Carl Hamacher and Hong Jiang. Comparison of Mesh and Hierarchical Networks for Multiprocessors. In *Proceedings of the 1994 International Conference on Parallel Processing*, pages 67–71, 1994.

[25] Govindan Ravindran and Michael Stumm. A Performance Comparison of Hierarchical Ring- and Mesh- Connected Multiprocessor Networks. In *Proceedings of the 3rd IEEE Symposium on High-Performance Computer Architecture*, 1997.

[26] William J. Dally and Brian Towels. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.

[27] W. Dally and C. Seitz. Deadlock-free Message Routing in Multiprocessor Interconnection Networks. *IEEE Transactions on Computers*, 36(5):547–553, May 1987.

[28] C. Carrión, R. Beivide, J. A. Gregorio, and F. Vallejo. A Flow Control Mechanism to Avoid Message Deadlock in k-ary n-cube Networks. In *Proceedings of the Fourth International Conference on High-Performance Computing*, 1997.

[29] C. Carrion, R. Beivide, J. A. Gregorio, F. Vallejo, and Dpto Electronica Y Computadores. Necessary and Sufficient Conditions for Deadlock-free Networks. *Technical Report Dpto. Electrdnica y Computadores. Available at: http:www.atc.unican.es*.

[30] V. Puente, J. Gregorio, and R. Beivide. SICOSYS: An Integrated Framework for Studying Interconnection Network Performance in Multiprocessor Systems. In *Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, 2002.

[31] Valentin Puente, Ramn Beivide, Jose Gregorio, J. M. Prellezo, Jose Duato, and Cruz Izu. Adaptive Bubble Router: A Design to Improve Performance in Torus Networks. In *Proceedings of the International Conference on Parallel Processing*, 1999.