Catnap: Energy Proportional Multiple Network-on-Chip

Reetuparna Das
University of Michigan
reetudas@umich.edu

Satish Narayanasamy
University of Michigan
nsatish@umich.edu

Sudhir K. Satpathy
University of Michigan
sudhirks@umich.edu

Ronald G. Dreslinski
University of Michigan
rdreslin@umich.edu

Abstract

Multiple networks have been used in several processor implementations to scale bandwidth and ensure protocol-level deadlock freedom for different message classes. In this paper, we observe that a multiple-network design is also attractive from a power perspective and can be leveraged to achieve energy proportionality by effective power gating.

Unlike a single-network design, a multiple-network design is more amenable to power gating, as its subnetworks (subnets) can be power gated without compromising the connectivity of the network. To exploit this opportunity, we propose the Catnap architecture which consists of synergistic subnet selection and power-gating policies. Catnap maximizes the number of consecutive idle cycles in a router, while avoiding performance loss due to overloading a subnet.

We evaluate a 256-core processor with a concentrated mesh topology using synthetic traffic and 35 applications. We show that the average network power of a power-gating optimized multiple-network design with four subnets could be 44% lower than a bandwidth equivalent single-network design for an average performance cost of about 5%.

1. Introduction

Energy proportional computing [1, 3, 11] requires that computing systems consume proportionally lower power when the computation demand is lower. For a processor to be energy proportional, its key components, such as the on-chip network must be energy proportional. As the number of cores increases, a high proportion of the processor's power will be consumed by its on-chip network. Borkar [6] argues that network power for a many-core die in the future could be as high as 150W if we naively scale current network implementations.

An on-chip network is energy-proportional if it consumes power that is proportional to the network demand and has insignificant impact on network latency. In a many-core system, even when all processor cores are actively computing, network demand may not always be near saturation. In fact, past studies have shown that real world applications exhibit bursty network traffic [10, 22]: a few phases that consume peak network bandwidth, and other computationally intensive phases that inject few packets into the network. Thus, to build an energy proportional many-core processor, it is important to design energy proportional on-chip networks.

Energy proportionality can be achieved through power gating unused network components, which reduces leakage power. At low network load, leakage power constitutes a dominant fraction of network power. Even at network saturation, we find that leakage power due to network components can be as high as 39% in a 256-core system. As the number of routers and their complexity increases to scale up the bandwidth for processors with hundreds of cores, the fraction of total power due to leakage is likely to increase. This leakage problem will only compound as the technology scales down to advanced deep-submicron nanometer regimes [5].

Runtime power gating has been successfully employed for various processor structures to reduce leakage power. However, solutions for power gating network components have been lacking. The problem is that, in a single network-on-chip (`Single-NoC`) design, even under low network load with only a few active flows, a majority of the routers in the network needs to be kept active to service those flows, significantly reducing the opportunities for power-gating. In fact, we find that power gating `Single-NoC` often results in significant performance penalty with little leakage power savings, because of frequent transitions between power states that often cause packets to wait for a power gated router to wake up. This is unfortunate, because applications tend to contain different phases with significantly varying network demands.

We observe that, unlike a single network-on-chip design, a multiple network-on-chip design [26, 27, 32] is more amenable to power gating, and therefore is an attractive solution for scaling network bandwidth. A `Multi-NoC` partitions wires and buffers into several subnetworks (subnets), and each node is connected to corresponding routers in all the subnets. `Multi-NoC` is more suitable for power gating because an entire subnet in a `Multi-NoC` can be turned off without compromising the connectivity of the network. To exploit this opportunity in `Multi-NoC`, we must overcome several problems, which we discuss below.

First, we must design a subnet selection policy that is able to expose long periods of idle time in a subnet to minimize the overheads of power gating and maximize its benefits. A node in a `Multi-NoC` is connected to several subnets, and therefore it can choose any one of the subnets to transmit a packet. Conventional round-robin (RR) or random subnet selection policies are inadequate, as their objective is to uniformly distribute the network load across subnets. Consequently, every subnet has to service some traffic, and as a result a majority of its routers needs to be kept active (similar to the issue we discussed for `Single-NoC`), squandering power gating opportunities.

Second, the subnet selection policy should be able to quickly adapt network bandwidth to an application's bandwidth demands and operate efficiently in the presence of bursty traffic. Real applications are often characterized by phases of varying network utilization. When network load starts to increase, the subnet selection policy must be able to utilize higher number of subnets and avoid performance loss. When network load starts to decrease, the subnet selection policy must be able to utilize fewer subnets so that the remaining subnets can be gainfully power gated.

Third, the power gating policy should also be to react quickly to congestion. This problem becomes particularly important for large processors with many cores. The power gating policy determines

ISCA13, Tel-Aviv, Israel.

when to wake up a router, and hence it must react quickly to congestion to avoid accruing wake-up delays from powered-off routers at each hop.

To address these problems, we propose the *Catnap* NoC architecture that employs a new *subnet-selection policy* and a new *power-gating policy*. Catnap's subnet selection policy enforces a strict priority between subnets during injection. A packet is injected into a higher-order subnet only if the current set of active subnets are getting close to *congestion*. This priority ordering ensures that at runtime only the required number of lower-order subnets are active, while higher-order subnets can be powered-off. It also ensures efficient operation under bursty traffic. After a burst, as the network load reduces, congestion in all the subnets reduces. Once the lower-order subnet's congestion goes below the congestion detection threshold, they are prioritized so that new packets are not injected into the higher-order networks.

To determine when a router and its associated links should be turned on/off, we discuss a *power-gating policy* that works synergistically with the Catnap's subnet-selection policy. Its objective is to maximize the sleep cycles while reducing frequent switches between power states and performance loss. In our design, a router in a subnet is turned off when its input buffers are empty for a predefined number of consecutive cycles *and* the congestion status of currently active subnets is all set to false. A power-gated router is woken up when either of these two conditions change.

Our proposed mechanisms rely on congestion detection. To achieve fast congestion detection, we discuss a *regional congestion detection* mechanism, which is the critical enabler of Catnap's subnet-selection policy and power-gating policy. A node detects congestion in a subnet if any node in its region detects local congestion in that subnet. A node determines its local congestion status of a subnet by examining that subnet's local router every cycle.

We investigated several solutions for locally detecting congestion in a subnet. We found that some of the seemingly promising local congestion metrics, such as the local packet injection rate, did not perform well. The reason is that the injection rate threshold for determining congestion varies significantly by the type of network traffic, which can change at runtime based on application characteristics. Congestion metrics, such as occupancy of injection queue (network interface queue) or average buffer occupancy of all ports of a router, did not perform well either. We found the maximum buffer occupancy of a local router to be the most effective local congestion metric, as it has the key advantage that its congestion threshold is independent of the network traffic pattern. Also, it incurs lower design complexity than the other alternatives we considered.

`Multi-NoC` is attractive even from a dynamic power perspective. For low bandwidth networks with fewer nodes, the overhead of duplicating control logic across multiple routers in different subnets could be expensive in terms of area and power. However, for high-bandwidth networks, a `Multi-NoC` design with multiple narrower networks is more efficient in terms of dynamic power than a bandwidth-equivalent `Single-NoC` design with a single wider network. The reason is that, beyond a datapath width, increasing the width of a router incurs a higher power cost than increasing the number of routers. We find that, at a higher datapath width, the latency of a crossbar becomes the bottleneck in the router pipeline. Therefore, a wider router needs to be operated at a higher voltage than a narrower router to achieve the same frequency. As power increases quadratically with respect to voltage, for high-bandwidth networks, the power of a single wider router is higher than the aggregate power of multiple narrower routers.

We evaluate 8x8 concentrated mesh for a 256-core system. Our evaluations show that Catnap's power-gating mechanism is effective in profitably power gating `Multi-NoC`'s network components for as much as 70% of execution cycles, while losing less than 2% performance for workloads with low network demand. However, for `Single-NoC`, we not only observe that there is only a negligible reduction in static power, but also that there is about a 10% performance loss for workloads with low network demand. When averaged over different multiprogrammed workloads, we find that the average network power of a Catnap `Multi-NoC` with four subnets (20W) is 44% lower than a bandwidth equivalent `Single-NoC` design (36W), while the average performance overhead is about 5%.

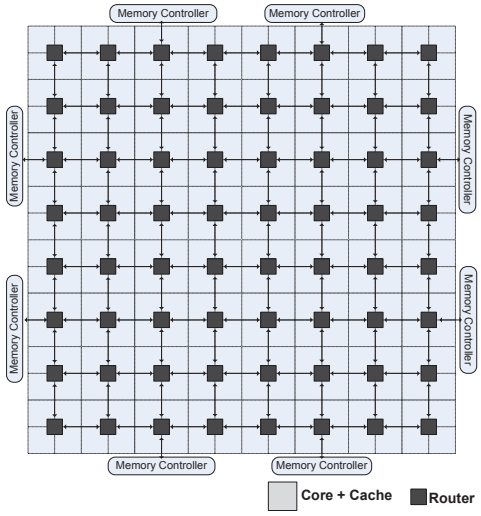

Figure 1. A processor with 256 cores connected by a single 8x8 concentrated mesh. Each router is connected to four tiles.

2. Background and Motivation

This section provides a brief background on packet-based network-on-chip (NoC). We discuss network power-scaling issues of a single physical network design, which are likely to motivate processor manufacturers to adopt a multiple-network design.

2.1 Packet-Based Network-on-Chip Architecture

Our baseline many-core processor with 256 cores and 8 memory controllers is illustrated in Figure 1. The on-chip network is organized as a grid of routers in an 8x8 concentrated mesh topology. A tile consists of a processor core, its private cache, and a slice of the shared last-level cache. Four tiles are connected to a single router through a shared Network Interface (NI) buffer. A router has five input ports: four ports to connect with the neighboring four routers, and one port to connect with the node's NI.

A data or a control message is communicated as one packet. Each packet consists of several smaller *fl*ow-control un*it*s called *flits*. Each flit travels hop by hop from one router to another until it reaches the destination router. The router we model is input buffered with virtual channels, uses wormhole switching, has a speculative two-stage pipeline [24], and performs look-ahead routing [12]. For an in-depth introduction to NoCs, we refer the reader to [9, 17].

2.2 Scaling On-Chip Network Bandwidth

Our target processor core count for this study is 256 cores. For sustaining performance the improvement of many-core processors, it is essential to maintain the current per-core bandwidth, although under tight power constraints [6]. To sustain today's 8 GB/s per-core bandwidth [15] in a 256-core processor, the bisection bandwidth of an 8x8 concentrated mesh described in Section 2.1 would have to

be 2 TB/s. Increasing the number of routers (from the 4x6 concentrated mesh used in current 48-core processor implementation such as Intel's SCC [15] to an 8x8 concentrated mesh) alone will not be sufficient to attain the required bisection bandwidth. Assuming a 2 GHz router, the link width has to be increased from 128 bits (used in Intel's SCC chip [15]) to 512 bits.

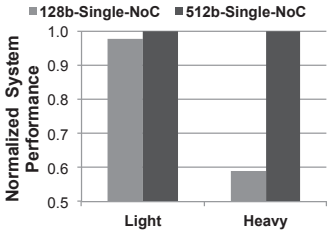

Figure 2. Performance of a 256-core processor with two different on-chip network bandwidths for two workloads with low network utilization (*Light*) and high network utilization (*Heavy*).

Figure 2 illustrates the impact of per-core network bandwidth on the performance of our target 256-core processor for two multi-programmed workloads: a *Light* workload, which has low network utilization, and a *Heavy* workload, which has high network utilization (please see Section 4 for evaluation methodology and Table 3 for details of applications used for each workload). We study two configurations: 1) a link width of 512 bits, which sustains today's per-core bandwidth and 2) an under provisioned system with a link width of 128 bits. We observe that an under provisioned 128-bit `Single-NoC` would result in almost 41% performance loss for *Heavy* workload compared to a 512-bit `Single-NoC`, which establishes the need to sustain today's per-core bandwidth.

Another alternative to scale bandwidth is to use multiple networks (`Multi-NoC`). In this work we study both `Single-NoC` and `Multi-NoC` network architectures for scaling network bandwidth. Several current processor implementations, such as Tilera [32] processor chip, already use multiple network-on-chip (`Multi-NoC`) design. Their main objective is to support different message classes and thereby ensure protocol-level deadlock freedom using a low-complexity solution. In addition to the above benefits, we argue in this paper that increasing the number of physical networks, instead of increasing the datapath width, is an attractive solution even from a power perspective as we scale to hundreds of cores.

Finally, a high-radix topology such as a flattened butterfly [19] could also be used to achieve higher network bandwidth. We only consider mesh topology in this paper, as it has a lower design complexity and has been used in several commercial processor implementations [15, 26, 27, 32]. Nevertheless, we believe that multiple physical networks would be advantageous even for a high-radix topology, as they could benefit from the power-gating optimizations discussed in this paper.

2.3 Multiple Network-on-Chip Design

We now describe how we extend the `Single-NoC` shown in Figure 1 to a `Multi-NoC` design that we analyze in this paper. In `Multi-NoC`, four tiles are connected to a single Network Interface (NI), which is the same as in a `Single-NoC`. The NI in a node is connected to several routers as shown in Figure 3; each router belongs to a different *subnet*. A processor core in a node stores a packet in that node's NI. When the packet reaches the head of NI, one of the subnet is selected (based on policies discussed in Section 3), and the packet is injected into that subnet. All the flits of a packet travel in the same subnet until they reach the destination.

A subnetwork in a `Multi-NoC` has the same topology as in a `Single-NoC`, but each subnet's datapath width is smaller than that of `Single-NoC`. We assume that the size of a packet for a message type remains the same across all the network configurations we

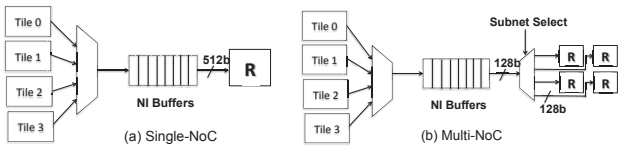

Figure 3. Organization of `Single-NoC` and `Multi-NoC`.

study. If the packet size is a constant, the number of flits per packet in a `Multi-NoC` with narrower subnets has to be higher than its bandwidth-equivalent wider `Single-NoC`. This is because a flit cannot be larger than the datapath width of a subnet.

Protocol deadlock is avoided by mapping dependent message classes to different virtual channels. Each router in a subnetwork is equipped with virtual channels. The messages which require point-to-point ordering can be mapped to one specific lower-order subnetwork which implements the same mechanism for point-to-point ordering used in a `Single-NoC`. Note that only few message classes (e.g. the control messages used for request forwarding) in directory coherence protocol require point-to-point ordering.

While studying various network configurations, we keep the number of wires and aggregate datapath width of all subnets constant. The total router buffer space is kept constant. In `Multi-NoC`, we divide the total buffer space equally among its subnets. But since the flits in a narrower subnet are proportionally smaller than the flits in a wider `Single-NoC`, buffer depth in terms of flits in each router is a constant in all the network configurations.

2.4 Achieving Energy Proportionality with Multi-NoC

Real-world applications tend to exhibit varied network demand. For example, Figure 2 illustrates that network bandwidth obtained from a 128-bit network is sufficient for *Light* application, however a 512-bit network is necessary for *Heavy* application. In addition, even individual applications exhibit phase behavior during runtime. They have some memory intensive phases that consume peak network bandwidth, necessitating a high-bandwidth network. But during computationally intensive phases, network bandwidth demand is much lower. During such phases of low network utilization, unused routers and links could be power gated to reduce leakage power and achieve energy proportionality.

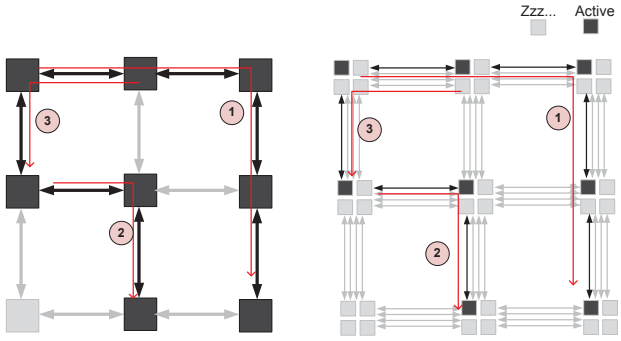

Figure 4. The left figure shows that all except one router is active in `Single-NoC` for a set of traffic flows (labelled as 1, 2, and 3). The right figure shows a bandwidth-equivalent `Multi-NoC` design with four subnets handling the same traffic flows. The routers and links in only one of the subnets is active, and the rest (28 out of 36 routers) are all in sleep state.

A single-network design, however, is ill suited for power-gating optimization, as illustrated in Figure 4. In our example, there are only three active traffic flows, but to support those flows, a majority of the routers and links in the network needs to remain active. Given the power cost of turning network components on and off, and the

performance cost incurred when packets are delayed to wake up the components from a sleep state, it is not generally fruitful to apply power gating even when the network demand is low. On the other hand, a `Multi-NoC` design provides excellent opportunities for employing power-gating optimizations. As shown in Figure 4, at low network load, only one subnet needs to be active, which allows us to power gate a majority of the routers (28 out of 36 routers).

3. Catnap: Power Gating for Multi-NoC

A multiple physical network design provides opportunities for fine-grained power management that are otherwise not available in a single-network design. This section discusses these new opportunities for power gating and proposes mechanisms to exploit them.

3.1 Power States and Power-Gating Costs

Runtime power gating of on-chip networks can be achieved by shutting off the power supply of an idle router and its associated links by turning off power switches (or sleep transistors) inserted between the V_{dd} line and the routers. In Catnap, a router can be in any one of three power states: *active, sleep, wake-up*. A router is said to be in active state when it is provided with a full supply voltage. It can transition into a sleep state in one cycle, in which it is completely turned off. However, there is a delay in waking up a router (`T-wakeup`) to charge the local voltage back up to V_{dd}. While transitioning from the sleep state to the active state, a router spends its `T-wakeup` cycles in the wake-up state.

While power gating can reduce leakage power of a component when it is in the sleep state, it incurs two costs. One is the performance overhead incurred when a packet has to be delayed to wake up a router. This cost depends on the `T-wakeup` delay, but it can be reduced by waking up a router before a flit reaches it. The second is the power cost that needs to paid to be switch a sleep transistor on and off, and charge the decoupling capacitance. This cost can be represented as the number of cycles (`T-breakeven`) a router needs to be in sleep state for a switch-off (active state to sleep state transition) to be worthwhile.

The two costs of power gating can be reduced if we can maximize the consecutive number of idle cycles for a router. To maximize consecutive idle periods and effectively exploit power-gating opportunity in `Multi-NoC`, we solve two important problems. The first problem is to design a *subnet-selection policy* for choosing a subnet to transmit a packet (Section 3.2). The second problem is to design a *power-gating policy* that determines when router components switch between different power states (Section 3.3).

3.2 Subnet Selection Policy

The objective of Catnap's subnet-selection policy is to *maximize the consecutive idle cycles* of the routers in order to avoid energy loss due to frequent state transitions and performance loss due to router wake-up delays. However, care must be taken to make sure that our policy does not over subscribe a subnet, which could result in a significant performance loss.

We find that a naive round-robin or random policy tends to evenly distribute traffic among the subnets, which causes each subnet to behave similarly to the example we discussed for `Single-NoC` in Figure 4. Thus, these policies do not expose long periods of idle cycles in routers that are necessary to benefit from power gating.

We propose a subnet-selection policy based on our observation that a subnet need not be activated until the current set of active subnets are getting close to congestion. To achieve this goal, Catnap enforces a strict priority ordering between various subnets. The NI at a router first attempts to inject a packet into the subnet-0. Only if it predicts that subnet-0 is getting close to congestion, it would inject its packets into the next subnet-1. Thus, if a lower-order subnet-l is free from congestion, it will always be preferred over all its higher-order subnets, subnet-h where $l < h$. From all the subnets that are close to congestion, the NI selects subnets using a round-robin policy.

When the network demand is high, higher-order subnets would be used for transmitting packets. When the network load is low, only the first few lower-order subnets would be used. Thus, Catnap's subnet-selection policy utilizes only as many subnets as are needed to meet the network demand of a workload's execution phase. Unlike round-robin and random policies, under low load, Catnap's policy exposes long periods of idle cycles in the routers of higher-order subnets, allowing them to be power gated. It achieves this goal while simultaneously preventing performance loss by detecting and avoiding congestion in a subnet early enough.

3.2.1 Detecting Congestion

To use our subnet-selection policy, a node in `Multi-NoC` relies on the ability to detect congestion in the subnets to avoid performance loss. A router's buffer in a subnet will start to fill up if its subnet is getting congested. Therefore, to detect congestion in a subnet at a particular node in the network, we propose a local congestion metric called the maximum buffer occupancy (*BFM*). Occupancy of a router's input buffer is the number of flits in that buffer. In Catnap, the NI at a node keeps track of the buffer occupancy of each router's input buffer. A node calculates BFM for a subnet as the maximum of the buffer occupancy of all the input ports of that subnet's router. If the *BFM* of a router is greater than a threshold, then that router's subnet is considered to be congested, and a local congestion status bit is set. For stability, this status is only reset after the BFM falls below a threshold. Thus, once a subnet is declared congested, it remains in that status for a few cycles.

The *BFM* congestion detection mechanism is local to a network node. It relies on the assumption that when congestion sets in any pocket of a subnet, the back pressure will build up in the subnet, and eventually reach the router that is attempting to inject a new packet into the subnet. However, the time to build up the back pressure could be so long that it may not be soon enough to avoid performance loss due to oversubscription to lower-order networks. This may cause intermittent latency spikes. Such scenarios frequently occur, especially when the traffic pattern is non-uniform across the subnet (e.g., applications with different network demands concurrently running on different nodes).

We can solve this problem if a node's NI has early knowledge of congestion in the routers of its neighboring nodes. To achieve this, we use a 1-bit OR network that collects the congestion status of all the routers in a region of a subnet. This bit value, which we refer to as the regional congestion status (`RCS`), can be read by all the routers in a region. In our design, we partition the 8x8 mesh subnet into four regions of 4x4 routers. The NI of a node sets its RCS if its local congestion status (`LCS`) is true which is determined based on the *BFM* of its local router. A node's NI detects congestion in a subnet if its `RCS` is true (that is, any one of that subnet's routers in its region is congested). A node's NI detects congestion for a subnet if either the local congestion status (`LCS`) is true (based on *BFM* of local router), or if the regional congestion status (`RCS`) is true (i.e., the OR-network's bit is set). The OR-network is architected as an H-Tree network. Section 4 discusses the power and latency details of our design obtained through SPICE analysis.

3.3 Power Gating Policy

Power-gating policy specifies the conditions under which a router (and its associated links) should be deactivated and activated. As we described in Section 3.1, the power-gating policy should be able to predict when there is likely to be a long enough idle period time at a router, and also be able to predict when a sleeping router should be activated to avoid performance loss.

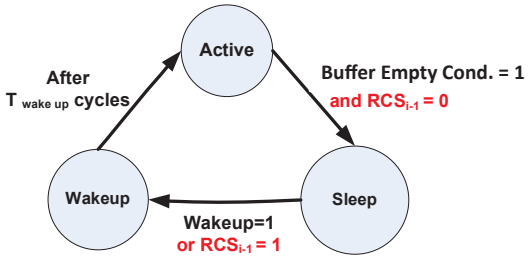

Figure 5. Router's states for Power Gating

Figure 5 shows the power state transitions for a router. We employ the following mechanism to determine when to deactivate a router. A router in a subnet S_h is unlikely to receive a flit if the regional congestion status of the immediate lower-order subnets S_l (where $l = h - 1$) is off. Under this condition, Catnap's subnet-selection policy would choose the lower-order network S_l to inject a packet. Therefore, our power-gating policy decides to switch off a router in a subnet S_h if the `RCS` of its immediately lower-order subnet S_l ($l = h - 1$) is off, *and* buffer empty condition is true. Buffer empty condition is true only if all of the router's buffers have been empty for a pre-defined number of consecutive cycles (four in our experiments). Note that our power-gating policy would keep the `0`th subnet always active.

To wake up a router of a subnet, Catnap's policy again relies on the RCS of its immediately lower-order subnet. A sleeping router in a subnet S_h is woken up and the power state of the router transitions to the wake-up state immediately when the RCS of its immediately lower-order subnet S_l ($l = h - 1$) is turned on. Also, when a router receives the head flit of a packet and determines the next hop during routing computation, it sends a `Wakeup` signal to the downstream router to active it. This allows us to hide part of the wake-up penalty (`T-wakeup`) when the RCS-based policy fails to activate the router in time before the flit arrives. This back-up policy to activate a router essentially leverages *look-ahead routing* [12] to hide the wakeup delay [21]. For a two-stage router a wake-up delay of up to 3 cycles can be hidden by sending a *wakeup signal* from an upstream router to a downstream sleeping router.

3.4 Alternative Policies for Local Congestion Status

We use *BFM* to determine the local congestion status of a router (Section 3.2.1). We arrived at this policy after evaluating several other policies that looked promising when we started our investigation. Now we describe those policies and explain why they failed (Section 6.4 presents a quantitative evaluation).

3.4.1 Injection Rate (IR)

The NI of a node can measure its injection rate, which is the number of flits injecting into the network over a period of time. Injection rate can be useful in selecting subnets, because it is a direct measure of the load that is being injected into the network. However, we found that the threshold for the injection rate at which congestion manifests varies for different traffic patterns. We found that the injection rate at which congestion manifests for uniform random traffic patterns is much higher than for an adversarial traffic pattern like transpose. Conservatively choosing a small threshold that provides acceptable performance for all traffic patterns ends up providing few opportunities for saving leakage power.

3.4.2 Average Buffer Occupancy (BFA)

Instead of computing the maximum buffer occupancy of a router, we tried average buffer occupancy. This policy also did not perform well, because when congestion manifests along only a few paths, then only a subset of the input ports' buffers would get filled up. The rest may be empty, which unnecessarily lowers the average buffer occupancy, and as a result we may not be able to detect the congestion.

3.4.3 Injection Queue Occupancy (IQOcc)

Instead of computing the injection rate at a node, we also tried a metric based on occupancy of injection queue [7]. This metric did not perform well either, because it was too slow to react to congestion. The injection queues get filled up only after a router's buffers are filled up. Also, buffer occupancy of a router's ports indicate the congestion of its neighbors and hence is more optimal than occupancy of injection queues.

3.4.4 Blocking Delay Based (Delay)

We also evaluated a policy in which a node detects congestion in a subnet when the average blocking delay per flit of that subnet's router exceeds a certain threshold. Average blocking delay-based congestion detection does not suffer from the problems that the other three metrics we discussed face. We can identify a threshold that works well for all traffic patterns and bursty traffic. However, this policy is expensive to implement in routers. An accurate implementation would require a counter per flit to measure the blocking delay, an adder to sum the blocking delays of all flits in the routers, and a shift register to divide it by sum of flits. To reduce this measurement overhead, this metric can be approximated by sampling only a few flits and computing a moving average of the blocking delays for only the sampled flits. We evaluated this sampling-based approach and found that the maximum buffer occupancy (*BFM*) that we use in our final design outperforms it.

Cores	256 cores @ 2 GHz, 64-entry instruction window, 2-wide fetch/issue/commit
L1 Caches	32KB/core, private, 4-way set associative, 64-byte block, 2-cycle latency, write-back, split I/D caches, 32-entry MSHR
L2 Caches	256KB/core, shared, 16-way set associative, 64-byte block size, 6-cycle bank latency, 32 MSHRs
Coherence	4-hop MESI directory protocol
Network	2GHz 2-stage router, 4 VCs/port, 4 flits/VC 8x8 mesh topology with 4 tiles/node, wormhole switched VC flow control, X-Y deterministic routing
Main Memory	Eight 4GB DRAMs, 80 cycle access latency 8 on-chip memory controllers (MCs), 4 DDR channels providing 16GB/s per MC, up to 16 outstanding requests for each core per MC,

Table 1. Processor configuration

4. Methodology

We describe three main aspects of our simulation infrastructure. First, we describe our cycle-level performance simulator which models a 256-core processor with single-network and multiple-network configurations. Second, we describe our router power model. Finally, we describe our SPICE simulations for determining the power and latency of components used in power gating.

4.1 Multi-Core Simulator

We use a cycle-level trace-driven multi-core simulator [10] to evaluate the performance and power of the 256-core processor described in Section 2. We use a front-end functional simulator based on Pin [23] to collect instruction traces from applications, which are then fed into a back-end cycle-level simulator. Configurations for processor core model, two-level caches, DRAM, and memory controllers are listed in Table 1.

Our cycle-level simulator is integrated with a detailed packet-switched on-chip network model. Our network model consists of a state-of-the-art two-stage, wormhole switched, 5-port router microarchitecture [24]. The routers operate at 2 GHz, and we assume link widths of 512 bits for `Single-NoC` and 128 bits for

Multi-NoC with four subnets. These parameters are chosen to provide a per-core bandwidth of 8 GB/s, which is similar to the bandwidth provisioned in the Intel's SCC chip [15]. Other network parameters that are common to both Single-NoC and Multi-NoC are listed in Table 1.

We use deterministic X-Y routing and virtual-channel flow control. We rely on virtual channels to prevent protocol-level deadlocks. We model all the coherence messages assuming a 4-hop MESI directory protocol, including closed-loop feedback between cores, caches, directories, memory, and network. For application workloads, messages consists of one-flit control packets and larger data packets (64-byte cache block size and 72-bit header). Single-flit packets account for about 60% of all packets communicated in our workload simulations. For synthetic workloads (consisting of uniform random, bit complement, and transpose traffic patterns), we assume 512-bit packet size.

We study the various congestion detection policies described in Section 3.4. We extensively experimented with many different thresholds for all these policies, but we present the results only for the threshold that performed well across all the traffic patterns. The best performing thresholds for various regional congestion detection policies were: *BFM*: 9 flits, *BFA*: 2 flits, *Delay*: 1.5 cycles, and *IQOcc*: 4 flits (injection queues have a capacity of 16 flits).

Regional congestion status is communicated to all the nodes in a region through a 1-bit OR network. To analyze the speed and power consumed by the 1-bit OR network, we use SPICE analysis. We derive the length of the wire by assuming an H-Tree network to interconnect the routers in a region (4x4 sub-grid). We estimated the propagation delay to be 2.7ns, which is 6 cycles at 2GHz. Thus the routers update and latch in their regional status flip-flops every 6 cycles. The dynamic switching energy of the 1-bit OR network is 8.7pJ.

4.2 Network Power Model

We use Orion 2 [18] to estimate network dynamic and static power. We assume a processor implemented in 32nm technology operating at a temperature of 25C. Each tile is 1.25mm long and the link between routers is 2.5mm long, resulting in a chip dimension of 20mm x 20mm. We model FIFO buffer and power-efficient matrix segmented crossbars with 2 input and 2 output segments for all routers. The NI for a router is shared between the tiles connected to the router, which is the case for both Single-NoC and Multi-NoC. The power due to additional control logic needed in Multi-NoC to connect the NI to multiple routers in different subnets, is also modeled.

We obtain the per-port load factor from our cycle-level simulation for synthetic and application workloads (for the results in Figure 7 we assume a per-port load factor of 0.5). We assume that the bit switching factor is 0.15. These values are fed into Orion 2 [18] to estimate dynamic power.

Recent work by Hayenga et al. [14] has found some inaccuracies in the Orion tool. The reported inaccuracies primarily affect the router area, power of SRAM array-based buffers and multiplexer-based crossbars. We do not utilize Orion's area model for this work. We also assume register-based FIFO buffers instead of SRAM array-based buffers. We further optimized the FIFO reads in the Orion model by using a circular queue instead of shifting all the FIFO registers for each read. Also, we assume constant buffer space across all our configurations (Single-NoC and Multi-NoC) and explore no buffering optimizations. Thus our conclusions are not sensitive to the buffer power model inaccuracies. Finally, we assume a matrix crossbar and avoid the inaccuracies in the multiplexer-based crossbar models in Orion.

In addition to dynamic and static power, we also report the percentage of cycles that could be profitably power gated (compensated sleep cycles). The benefits quantified by this metric are independent of the Orion power model.

4.3 Power Gating Model

We did SPICE simulations to estimate router wake-up delay and the energy overhead of switching the sleep transistor on/off. We modeled the supply rail using capacitance and inductance, and the router port was modeled as a current source. The decoupling capacitance used is proportional to the power drawn for different router configurations. From our analysis, we determined the wake-up delay to be equal to 5.1ns for a 128-bit router (T-wakeup =10 cycles). Out of 10 cycles, 3 cycles are hidden by look-ahead routing [21]. We also calculated the energy overhead due to switching the sleep transistor on/off and charging decoupling capacitance. The energy overhead was found to be equivalent to 12 cycles worth of leakage energy for a router (i.e., the break-even point T-breakeven = 12 cycles). We assume the buffer empty condition is set only after router's buffers have been empty for T-idle-detect = 4 cycles.

5. Characterization of Multiple Networks

Processor implementations have started to employ Multi-NoC designs [26, 27, 32] as a low-cost alternative to scale bandwidth and also support different message classes. In this section we characterize the performance and power consumption of Multi-NoC compared to Single-NoC without considering power gating. While Multi-NoC does incur additional costs that are absent in Single-NoC, we identify several performance and power benefits to using Multi-NoC that could offset those costs, especially when we consider a processor with 256 cores or more. Later in Section 6, we evaluate our novel power-gating optimizations, which further tilt the scales in favor of Multi-NoC designs as we scale the number of cores.

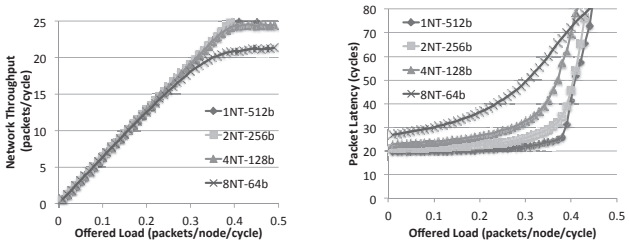

Figure 6. Performance of Single-NoC compared to several Multi-NoC designs.

5.1 Performance Characteristics

Figure 6 shows the network throughput latency for Single-NoC and different Multi-NoC designs at near saturation (that is, we assume a per-port load factor of 0.5). It can be seen that increasing the number of subnetworks beyond four results in some throughput loss. Average packet latency also increases by a few cycles with the increase in the number of subnets.

The main reason for this performance loss in Multi-NoC is due to the increase in the number of flits per packet. Number of flits per packet increases with the number of subnets, because we keep the aggregate link width constant across all the network designs to provide a fair comparison. For example, in a 8NT-64b Multi-NoC, the number of flits per packet is 8, whereas in 1NT-512b network the number of flits per packet is 1.

An increase in the number of flits per packet could affect network performance in two ways. The first way is due to constraints imposed by wormhole switching, which requires that all the flits of a packet be kept together in the network. Due to this constraint, an increase in flits per packet tends to increase the chance of congestion, which leads to some throughput loss for a Multi-NoC design

with a large number of subnets. The loss in throughput could also affect the network latency at high offered load.

The second way an increase in the number of flits per packet could affect network performance is to due to increase in serialization latency. The serialization latency of a packet is the number of cycles that elapses between the arrival time of head and tail flits at the destination node under zero load, which increases with the increase in flits per packet. An increase in serialization latency affects packet latency noticeably at low network load (Figure 6 (b)). However, at high network load, an increase in serialization latency cost is not significant, as it constitutes only a small fraction of the total packet latency.

Fortunately, some of the above performance overhead is offset by two performance advantages that `Multi-NoC` enjoys over `Single-NoC`. First, a higher number of subnets improves performance by reducing head-of-line blocking. Second, a wider `Single-NoC` is not optimal while communicating smaller control packets. For instance, flit size in a 1NT-512b network is much greater than a typical control packet size (72 bits), leading to internal packet fragmentation that causes poor network utilization.

We find that for our 256-core processor (Section 4), a `Multi-NoC` with four subnets provides similar throughput as a `Single-NoC`. Hence, in the rest of the paper we use four subnets for `Multi-NoC` design.

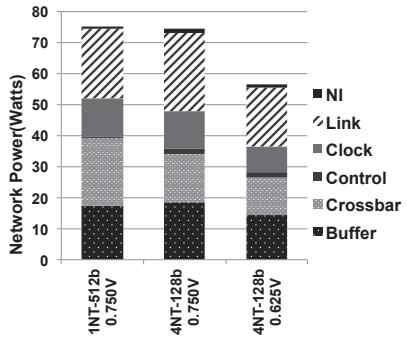

Figure 7. Network power for `Single-NoC` compared to `Multi-NoC`, with and without voltage scaling.

5.2 Power Characteristics

We find that the power of a `Multi-NoC` design compares favorably with that of an equivalent `Single-NoC`, as shown in Figure 7. We analyze power consumption at near saturation due to six main network components [18, 31]: 1) *Buffers*, 2) *Crossbar*, 3) *Control Logic*, 4) *Clock*, 5) *Link*, and 6) *Network Interface (NI)*.

The aggregate buffer sizes remain constant across `Single-NoC` and `Multi-NoC`, and therefore buffer power is almost the same across these designs. While `Multi-NoC` incurs additional overhead due to duplicated control logic and an increase in link lengths due to layout complexity, it could be offset by the aggregate power reduction due to the reduced complexity of narrower crossbars and clock distribution.

The control logic for buffers, routing, and arbitration needs to be replicated across the routers in different subnets. Fortunately, the overhead due to control logic does not significantly offset the benefits of `Multi-NoC`, as the power and area consumption of control logic is relatively cheap (<4% of total router power) compared to other components ([30], TRIPS [26], RAW [27]).

In addition to control logic overhead, link power could increase because the links from a router need to cross over the other routers within a node. However, links from different subnets can be routed through different metal layers to handle crossover points. The number of crossover points can be reduced using wire turns at routers and better router layout. We did a layout analysis for a `Multi-NoC` design with four 128-bit wide subnets and found that the link power could increase by about 12% compared to a `Single-NoC` with 512-bit links.

We found that for large data-path widths (128-bit width and beyond), the power due to a single large crossbar (512-bit) is higher than the power due to multiple smaller crossbars (128-bit). Overall, we found that the super-linear reduction in the complexity of router components, such as crossbar and clock, offsets the overheads due to control logic and link power in `Multi-NoC` compared to `Single-NoC`.

Design	Router width (bits)	Frequency (GHz)	Voltage (V)
`Single-NoC`	**512**	**2.0**	**0.750**
`Single-NoC`	512	1.4	0.625
`Multi-NoC`	128	2.9	0.750
`Multi-NoC`	**128**	**2.0**	**0.625**

Table 2. Frequency and voltage of 512-bit and 128-bit routers used in `Single-NoC` and `Multi-NoC` designs, respectively. Highlighted rows indicate the configuration we use in our evaluation.

Furthermore, narrower routers in `Multi-NoC` could be operated at a lower voltage than a single wider router in `Single-NoC`. To understand this benefit, we synthesized the circuit of arbitration and matrix crossbar stages for 32nm technology. We found that the critical path delay of the router is determined primarily by the crossbar for routers that are 256 bits or wider. Thus the critical path delay reduces as the router datapath width decreases from 512 bits (`Single-NoC` design) to 128 bits (`Multi-NoC` design). As a result, a narrower 128-bit router can be clocked at a lower voltage than a wider 512-bit router to attain the same frequency (2 GHz). The frequency and corresponding operating voltages for 128-bit and 512-bit routers are shown in Table 2. To achieve our target frequency of 2 GHz, a 512-bit router needs to be operated at 0.75V, whereas a 128-bit router can achieve the same frequency at a lower voltage of 0.625V. When we assume a lower voltage (0.625V) for a 128-bit router, we observe that `Multi-NoC` could provide significant dynamic power savings over `Single-NoC` (third stack bar in Figure 7).

6. Evaluation of Power-Gating Enabling Policies

We demonstrate that when power-gating optimization is applied to `Single-NoC`, it can hardly reduce static power and also incurs a significant performance loss. On the other hand, Catnap's power-gating policies are extremely effective for `Multi-NoC`, where we observe long periods of idle time in the higher-order subnets. We discuss these results for both application and synthetic workloads.

We also show that Catnap's synergistic subnet selection and power-gating policy, based on Buffer-Max (BFM) regional congestion detection, outperforms naive round-robin subnet selection and other policies that we considered in Section 3.4.

6.1 Network Configurations Studied and Metrics

We study the bandwidth-equivalent `Single-NoC` and `Multi-NoC` designs for a 256-core processor that we described in Section 4.1. `Single-NoC` is 512 bits wide (`1NT-512b`). `Multi-NoC` has four 128-bit subnets (`4NT-128b`). We study `Single-NoC` and `Multi-NoC` with and without power gating. When we do not employ power-gating for `Multi-NoC`, we use round-robin scheme for subnet selection.

Configurations with power gating enabled are indicated with the `PG` suffix. When we consider power gating for `Single-NoC`, we use a power-gating policy similar to the one discussed by Matsutani et al. [21]. It is the same as the power-gating policy illustrated in Figure 5, but without our regional congestion status (RCS) conditions.

Mix									avg. MPKI
Light	applu (32)	gromacs (32)	deal (32)	hmmer (32)	calculix (32)	gcc (32)	sjeng (32)	wrf(32)	3.9
Medium-Light	gromacs (32)	deal (32)	gobmk (32)	wrf (32)	h264ref (32)	sphinx (32)	applu (32)	calculix (32)	7.8
Medium-Heavy	cactus (32)	deal (32)	calculix (32)	hmmer (32)	namd (32)	sjas (32)	gromacs (32)	sjeng (32)	11.7
Heavy	sjas (32)	astar (32)	mcf (32)	sphinx (32)	tonto (32)	tpcw (32)	deal (32)	hmmer (32)	39.0

Table 3. Benchmarks used to construct four multiprogrammed workloads for our 256-core processor. Numbers in parenthesis indicate the number of application instances that we used to construct a workload. The last column lists the average Misses-Per-Kilo-Instruction (MPKI) per core, which is indicative of the network load for the workloads. Average MPKI for a benchmark is calculated as the sum of its L1-MPKI and L2-MPKI.

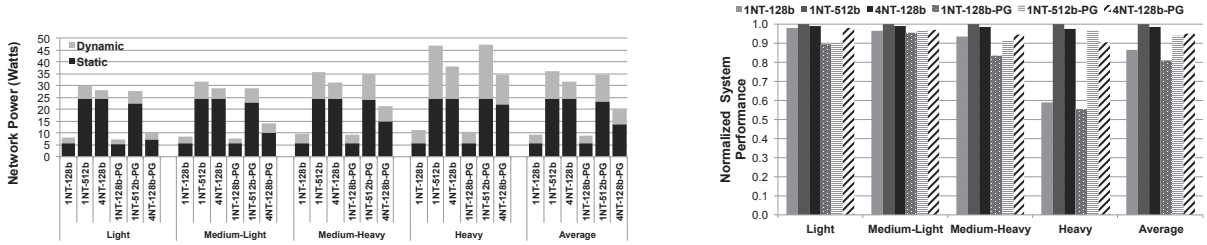

Figure 8. Network power (left) and processor performance (right) for application workloads.

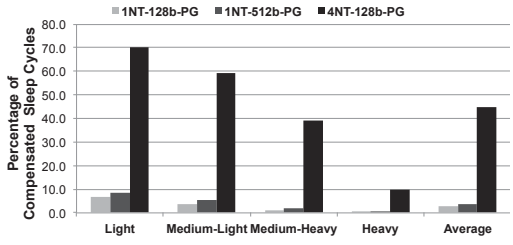

Figure 9. Compensated Sleep Cycles.

Also for the `Multi-NoC` baseline, we use the same power-gating policy [21] (without the RCS conditions) along with round-robin (RR) scheduler for subnet selection. These two baseline power-gating schemes are compared to our `Multi-NoC` design that uses the RCS-based power-gating policy shown in Figure 5. We use our best-performing RCS metric called Buffer-Max (BFM) in all our experiments except when we compare different congestion metrics (Section 6.4).

In addition to reporting absolute power numbers, we also quantify the power-gating opportunity using *Compensated Sleep Cycles (CSC)* [16]. CSC is equal to the sum of idle cycles minus the break-even cycles (`T-breakeven`) accounted for every sleep period. We present CSC as a percentage of total cycles elapsed in an execution, which is the fraction of time when routers and links do not incur any static power due to leakage.

6.2 Application Workloads

We study a diverse set of 35 applications comprising scientific, commercial, and desktop applications drawn from SPEC CPU2006, SPLASH-2, and SpecOMP benchmark suites, and four commercial applications (`sap`, `tpcw`,`sjbb`, `sjas`). We use Simpoint [23] to select representative execution phases for all our applications except commercial traces. Commercial applications were run natively on Intel servers and representative execution traces were collected using an Intel propriety hardware tracer.

We study four representative workloads, as listed in Table 3. The four workloads vary in terms of their network demand, ranging from *Light* to *Heavy* as indicated by the average Misses Per Kilo Instruction (MPKI) in the last column.

Figure 8 shows the network power (left) and processor performance (right) for `Single-NoC` and `Multi-NoC`, with and without power gating. We also include results for an under-provisioned 128-bit `Single-NoC` to illustrate the need for a high-bandwidth network in a 256-core processor. Performance is normalized to `Single-NoC` without power gating.

Static power for `Single-NoC` and `Multi-NoC` is about the same (25W) without power gating across all workloads. With power gating, we were successful in significantly reducing this static power for `Multi-NoC`, but not for `Single-NoC`.

Power gating for networks is understandably more effective at relatively lower network demand levels. For *Light* workload, power gating for `Multi-NoC` saves as much as 70% of static power for a performance loss of less than 2% compared to `Single-NoC` without power gating. We also observe that power gating is ineffective for `Single-NoC` even at low load, as it loses as much as 10% performance for only a negligible reduction in static power. The performance loss is due to the fact that `Single-NoC` exposes only short periods of idle cycles, which causes the routers to switch frequently between on and off states, resulting in performance loss with little power savings. In terms of total power, for *Light*, baseline `Single-NoC` with power gating consumes nearly 28W, whereas `Multi-NoC` with power gating consumes only 7.25W of total network power.

As one would expect, effectiveness of power gating reduces as the network load increases. For *Heavy*, power gating in `Multi-NoC` saves about 10% of static power. However, `Multi-NoC` is still significantly more power efficient (34.5W) than `Single-NoC` (46.8W), as it consumes lesser dynamic power due to voltage scaling (Section 5.2).

Assuming power gating, we observe that `Multi-NoC` consumes only about 20W of total network power on average, whereas `Single-NoC` consumes about 36W, which constitutes a 44% reduction in network power. The performance loss of `Multi-NoC` with power gating compared to `Single-NoC` *without* power-gating is about 5% on average. This performance loss could be reduced, if necessary, by reducing the aggressiveness of Catnap's power-gating optimization by adjusting the threshold used for regional congestion detection.

The percentage of compensated sleep cycles (CSC) (Section 6.1) shown in Figure 9) also establishes the difference in the effectiveness of power-gating optimization between `Single-NoC` and `Multi-NoC`. For *Light*, routers could be profitably power gated for as many as 70% of execution cycles.

6.3 Synthetic Workloads

We analyzed the performance and power of `Single-NoC` and `Multi-NoC` with and without power gating for synthetic traffic patterns. `Single-NoC` without power gating is our baseline refer-

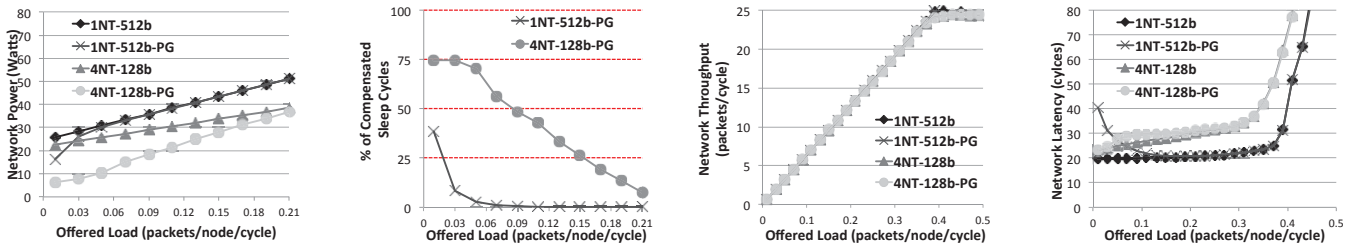

Figure 10. Power and performance of `Single-NoC` and `Multi-NoC`, with and without power-gating for uniform random traffic. (a) Network Power (b) Compensated Sleep Cycles (c) Network Throughput and (d) Network Latency

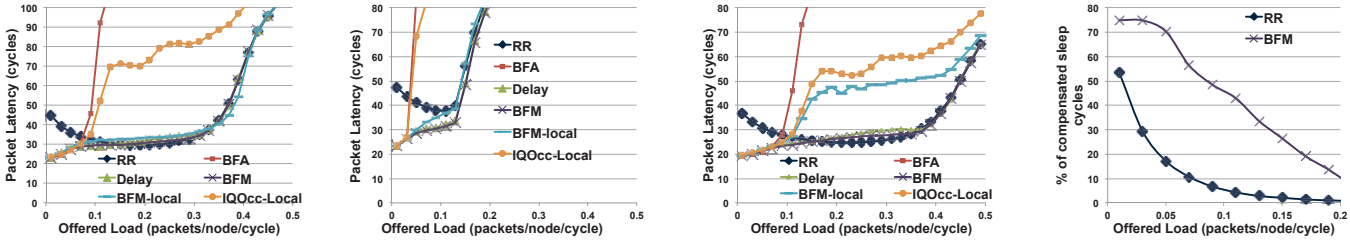

Figure 11. Performance of various local congestion metrics (Section 6.1) for (a) Uniform random (b) Transpose and (c) Bit Complement traffic patterns. Rightmost graph shows compensated sleep cycles while using RR (naive) and BFM (best) policies.

ence. Figure 10 presents this result for the uniform traffic pattern. We also studied transpose and bit-complement traffic patterns, but we omit those results in the interest of space. Our conclusions remained the same for those traffic patterns as well.

Our first observation is that our power-gating optimization is effective for `Multi-NoC`, but fails for `Single-NoC` due to the reasons we discussed in Section 2.4. Consider a light network load of 0.03 packets/node/cycle. Even at this low load, `Single-NoC` exposes only 10% of CSC, whereas `Multi-NoC` exposes about 74% (Figure 10(b)). Higher CSC translate to a significant reduction in static network power for `Multi-NoC`. Figure 10(a) shows that, at low load, the total network power of `Multi-NoC` is only 7.8W, as opposed to 24.1W for `Single-NoC` when we apply power gating.

As one would expect, the percentage of CSC reduces as the network load increases. As a result, at high load (after 0.20 packets/node/cycle or more), power benefits from `Multi-NoC` are primarily due to a reduction in dynamic power.

Our next observation is that the network performance of `Multi-NoC` does not suffer significantly due to Catnap's power-gating optimizations. Figure 10(c) shows that the network throughput of both `Single-NoC` and `Multi-NoC` is largely unaffected due to our power-gating optimizations. The reason is that, at saturation, network components are rarely turned off, which avoids any wake-up penalties due to power gating.

However, we observe a noticeable increase in average packet latency due to power gating. For `Single-NoC` in particular, we observe that power gating could lead to a significant increase in average packet latency. This penalty is severe, especially at low load for `Single-NoC` due to the higher wake-up penalties caused by frequent switching between active and sleep states. At high loads, network components in `Single-NoC` are rarely ever turned off. As a result, the negative impact of power gating on packet latency for `Single-NoC` reduces with an increasing network load. Thus, we conclude that power gating is ineffective for `Single-NoC` due to its prohibitive performance overhead, and because it does not result in significant power savings.

`Multi-NoC`, on the other hand, performs significantly better with power gating and also saves significant network power. While packet latency could increase with the network load due to power gating, fortunately, at high network loads, applications tend to exhibit higher memory-level parallelism, where they are more sensitive to network throughput and are affected to a lesser degree by an increase in network latency.

6.4 Analysis of Congestion Metrics

In Section 3, we discussed Catnap's synergistic subnet section and power-gating policy based on regional congestion detection. A naive approach would be to use a round-robin (RR) policy for subnet selection and use the same power-gating policy we assume for `Single-NoC` [21]. Figure 11 compares the performance of RR with the various regional congestion status (RCS) based subnet selection and power-gating policies that we discussed in Section 3.4 for `Multi-NoC` (*4NT-128b-PG*).

First, we observe that average packet latency is significantly higher when we use RR with power gating. RR policy equally distributes network load across different subnets, which causes each subnet to behave like a `Single-NoC` operating a lower load. As a result, it suffers from the same performance issues as that of `Single-NoC` when we apply power gating. Also, similar to `Single-NoC`, the percentage of CSC is significantly lower than that of our best performing RCS policy which uses Buffer-Max (BFM) as the local congestion metric.

Let us now consider other local congestion metrics. First, we observe that average buffer occupancy metric (*BFA*) and injection queue occupancy metric (*IQOcc*) result in lower throughput than even the baseline RR policy, because they are not capable of detecting congested paths quickly enough. Second, we observe that both *Delay* and our final policy, based on maximum buffer occupancy per port *BFM* provide almost similar performance. Also, we found that they are effective in exposing higher percentage of CSC (Figure 11(d)). We chose to use BFM, because of its lower design complexity when compared to *Delay*.

Finally, we observe that RCS-based policy performs better than a subnet selection and power-gating policy that only uses the local BFM status (*BFM-local*). *BFM-local* cannot detect congestion as early as RCS-based BFM, especially for non-uniform traffic patterns (Figure 11(c)), which could overload an already congested lower-order subnet. This justifies our design choice to employ a 1-bit region congestion status OR network.

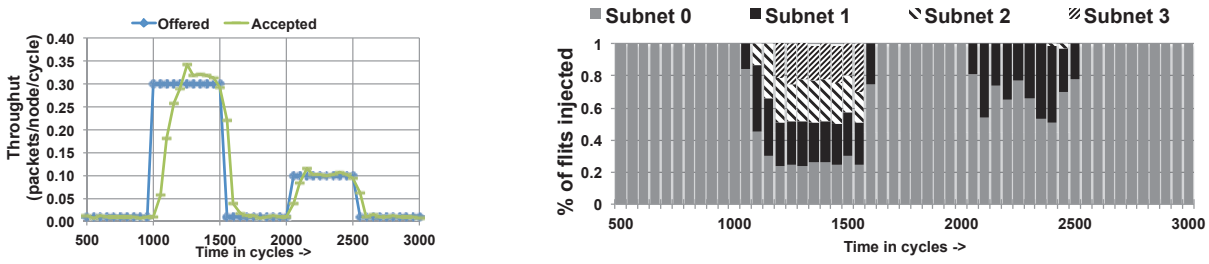

Figure 12. `Multi-NoC` throughput for bursty traffic. (a) Offered and accepted throughput of the network with time, and (b) Utilization of subnets with time.

Of particular interest is how an intuitive policy based on packet injection rate (*IR*) fails when we use it for subnet selection. Figure 13 shows the average packet latency for `Multi-NoC` when we use *IR* policy to select a subnet during packet injection. We do not consider power gating for this experiment.

We experimented with various IR thresholds for detecting congestion at the local node. We observe that we can choose a threshold as high as 0.20 packets/node/cycle without any loss in performance. However, for the non-uniform transpose traffic, which saturates the network more quickly, we require a much lower threshold of 0.08 packets/node/cycle to prevent performance loss. However, a low threshold would diminish the potential for power savings. Thus, we would need be able to adjust the threshold based on the traffic characteristics if we had chosen IR as our congestion metric. Fortunately, for our BFM metric, a constant threshold is adequate to expose most of the power-gating opportunities and also does not cause significant performance degradation.

6.5 Ramp-Up and Decay Time with Bursty Traffic

We tested the robustness of our BFM subnet and power-gating policy in quickly adapting to changes in network load. Our policy should be capable of quickly reacting to any increase in network load and be able to open up higher-order subnets. Also, it should be able to quickly deactivate the subnets when the traffic dies down. Figure 12(a) shows the ramp-up and decay time for bursty traffic for our `Multi-NoC` design with power gating.

We simulate two bursts. The offered load for the network is increased from 0.01 to 0.30 packets/node/cycle at 1000 cycles and retained at that level until 1500 cycles to emulate the first burst. We sample network throughput every 50 cycles and plot it in Figure 12(a). Accepted throughput catches up with the offered throughput in 200 cycles. Also, they can quickly ramp-down when the bursty period ends. Figure 12(b) shows subnet utilization for `Multi-NoC`. Before and after the burst, subnet-0 is active. During the burst, our BFM policy is able to quickly detect congestion in the lower-order subnets and activates all the subnets to equally spread the network load among them during the bursty period. When offered load is increased from 0.01 to 0.10 packets/node/cycle for a second burst, at 2000 cycles, only two subnets (subnet-0 and subnet-1) are activated, because the offered load requires only two subnets to satisfy the needed bandwidth.

6.6 Discussion

In this section we discuss the limitations of our Catnap design and its suitability for different processor configurations. The `Multi-NoC` design is attractive for large processors with high network bandwidth requirements. However, as the number of cores reduces, the network bandwidth requirement reduces, and fewer subnets are necessary to satisfy the processor's per-core bandwidth requirement. The opportunities for power gating is proportional to the number of subnets. Hence, the benefits of our proposed policies are lower for smaller processors.

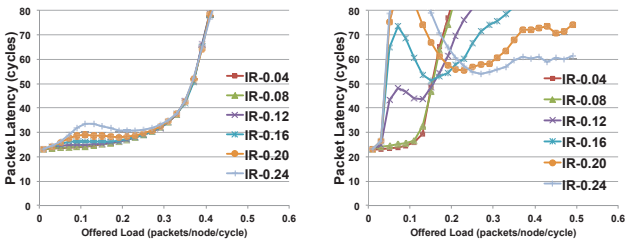

Figure 13. `Multi-NoC` performance with IR policy for uniform-random and transpose traffic patterns.

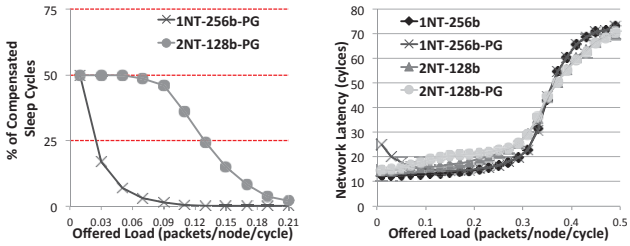

Figure 14. Comparison of `Single-NoC` and `Multi-NoC` with uniform random traffic for 64-core processor: (a) Compensated sleep cycles and (b) Network latency

We believe our design is profitable for processors with 64 or more cores. It is likely that for processors with fewer than 64 cores, there will be no need for more than one network. For a 64-core processor with a 4x4 concentrated mesh, a `Single-NoC` design will be 256 bits wide (to sustain a per-core bandwidth of 8 GB/s) and an optimal `Multi-NoC` design will have two 128-bit subnets. Note, we use concentrated topology because concentration is a simple and effective way to reduce network latency and power [2, 9].

Figure 14 shows the compensated sleep cycles (CSC) and network latency, with uniform random traffic pattern, for the 64-core processor configuration. We observe 50% compensated sleep cycles for `Multi-NoC`, but only 17% for `Single-NoC` at a network load of 0.03 packets/node/cycle. As a comparison, for the same network load, for 256-core processor with 4-subnet `Multi-NoC`, we observed nearly 74% CSC. The lower power gating benefits stem from the fact that we could not afford more than 2 subnets for the 64-core `Multi-NoC` design since that would cause performance loss (Section 5.1). We conclude that our proposed policies are effective for a processor with 64 cores, and that the benefits of a `Multi-NoC` design with power gating are higher as the number of cores increases.

7. Related Work

We discussed the Catnap architecture to exploit opportunities for power-gating in multiple networks using a regional congestion de-

tector. In this section we relate our contributions to prior studies on power-gating for NoCs, Multi-NoC architectures, and congestion detectors.

7.1 Power Gating for On-Chip Networks

Recent studies have explored several power-gating optimizations for NoC with a single physical network [8, 20, 21, 25]. Matsutani et al. [20, 21] reduced the performance overhead due to wake-up delay of routers by leveraging look-ahead routing and using fine-grained power gating of router ports. Chen and Pinkston [8] proposed a separate bypass path in routers to allow powered-off routers to transmit packets. Samih et al [25] proposed router-parking, which proactively power gates routers associated with inactive cores and uses adaptive routing to retain connectivity between active routers.

These prior works aim to exploit power-gating opportunities in a Single-NoC. We used Matsutani et al [21]'s mechanism to study power-gating optimization for our baseline Single-NoC architecture (Section 6.1). However, as we discussed in Section 2.4, opportunities for power-gating in Single-NoC are limited as it is a challenge to expose long periods of consecutive idle cycles. Intermittent flow of packets require most of the routers in Single-NoC to be active to retain connectivity. On the contrary, Multi-NoC provides superior power-gating opportunities as an entire set of routers in a subnet can be power-gated while retaining full connectivity between nodes through other active subnets.

Nevertheless, power-gating mechanisms for Single-NoC could potentially complement Catnap's power-gating policies for Multi-NoC as they could be used to improve the effectiveness of power-gating in a subnet of our Multi-NoC design.

While HPC-mesh [7] considered power gating for a multiple network topology, it was a preliminary study with several issues, some of which are enumerated below.

- HPC-mesh did not quantify the power-saving benefits of power-gating by comparing a topology with and without power gating. No performance overhead due to power gating was reported.
- HPC-mesh relied on a global congestion detector, in which every router had instantaneous knowledge of the number of flits in the injection queues of all routers in each cycle. A subnet is considered to be congested if this global sum is greater than 200 flits [7]. As we showed in Figure 11, a more practical design for a 256-core processor that uses injection queue occupancy as congestion metric performs poorly (*IQOcc-Local*).
- HPC-mesh would turn off a router as soon as its buffers are empty, which could lead to significant performance loss if one assumes a practical router wake-up delay (10 cycles in our experiments, determined through SPICE simulations) for our 128-bit router operating at 2 GHz. HPC-mesh assumed 3-cycle wake-up delay, 2 cycles of which is hidden by overlap with arbitration and crossbar transfer stages.
- The policy for HPC-mesh would open up all the subnets and load balance them as soon as the first subnet is congested. This is not efficient for medium-demand workloads that require a bandwidth of two or three subnets.

This paper addresses the above issues. We also characterize the benefits, costs, and limits of a Multi-NoC architecture compared to its bandwidth equivalent Single-NoC. For a larger high-bandwidth network, we demonstrate significant dynamic power benefits for Multi-NoC over Single-NoC by applying voltage scaling.

7.2 Multiple On-Chip Networks

Tilera [32], TRIPS [26], and RAW [27] used multiple networks instead of virtual channels to isolate different message classes (e.g., instruction operands, coherence traffic, etc.) in order to provide protocol-level deadlock freedom and quality of service. Volos et al. [29] further argued for specializing subnets in a Multi-NoC for each message class (e.g., data versus control) to improve efficiency. Separating traffic into different subnets based on their message type could lead to load imbalance across subnets. Furthermore, even with multiple networks, we find that virtual channels are necessary for exploiting peak bandwidth supported by the network. Therefore, in our Multi-NoC design we use virtual channels to ensure protocol-level deadlock freedom. It avoids load imbalance issue and also helps attain close to peak bandwidth in each subnet.

None of these prior studies have exploited power-gating opportunities or observed dynamic power-scaling advantages in a Multi-NoC design.

7.3 Congestion Detection

Congestion detection has been used for adaptive routing [13] and injection throttling [4, 28] to improve network throughput. Our congestion detection mechanism is customized for aiding power gating in multiple networks, for which we find that a regional congestion detection based on maximum buffer occupancy performs well.

For efficient adaptive routing, it is important to predict congested paths [13], which requires a more complex mechanism than predicting whether an entire subnet is congested or not. For example, RCA [13] predicts congested paths by frequently communicating the local congestion status of each port of a router to all of its neighboring routers. In contrast, we employ a simpler 1-bit OR network which is set when any of the router's local congestion status is true. For injection throttling in off-chip multiprocessor networks, previous work [4, 28] used a global congestion detector. We instead used a regional congestion detector which can detect congestion in a subnet more quickly so that higher order subnets can be activated in time to avoid a performance loss.

8. Conclusion

Energy proportional computing requires computing systems to pay proportionally lower power costs when the computation demand is lower; it can be achieved by power gating unused components. Unfortunately, power gating interconnects is a fundamentally difficult problem because of its distributed nature. The problem is that, in order to provide connectivity between all nodes, routers incur frequent mode transitions between on and off states even at low network load. The overheads of power gating due to frequent mode transitions make it unprofitable for interconnects.

Multiple networks are an attractive solution to scale on-chip network bandwidth as the number of processor cores scale up. We observe that a Multi-NoC design is more amenable for power gating than a Single-NoC design. We discussed the Catnap architecture that employ regional congestion detection to effectively exploit the power-gating opportunities in Multi-NoC. We also characterized the power and performance of Multi-NoC, and showed that for high-bandwidth networks, multiple networks consume lower dynamic power than single networks if we assume voltage scaling.

We find that a Multi-NoC design with Catnap power-gating optimizations consumes about 44% less power than a bandwidth-equivalent Single-NoC design, which provides compelling evidence that future processor implementations should use Multi-NoC. While our study focused on a most widely used topology (concentrated mesh), further study is required to demonstrate similar benefits for other topologies.

9. Acknowledgements

We thank the Akhilesh Kumar, Mani Azimi and the anonymous reviewers for their input. This research is supported by the National Science Foundation under the awards CCF-1256203 and CAREER-1149773.

References

[1] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks," in *ISCA*, 2010.

[2] J. D. Balfour and W. J. Dally, "Design tradeoffs for tiled cmp on-chip networks," in *ICS*, 2006.

[3] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *IEEE Computer*, 2007.

[4] E. Baydal, P. Lopez, and J. Duato, "A family of mechanisms for congestion control in wormhole networks," *IEEE Trans. Parallel Distrib. Syst.*, 2005.

[5] S. Borkar, "Design challenges of technology scaling," *Micro, IEEE*, 1999.

[6] ——, "Thousand core chips: a technology perspective," in *DAC-44*, 2007.

[7] J. Camacho and J. Flich, "Hpc-mesh: A homogeneous parallel concentrated mesh for fault-tolerance and energy savings," in *ANCS-7*, 2011.

[8] L. Chen and T. M. Pinkston, "Nord: Node-router decoupling for effective power-gating of on-chip routers," in *MICRO-45*, 2012.

[9] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2003.

[10] R. Das, O. Mutlu, T. Moscibroda, and C. Das, "Application-Aware Prioritization Mechanisms for On-Chip Networks," in *MICRO-42*, 2009.

[11] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *ISCA*, 2007.

[12] M. Galles, "Scalable pipelined interconnect for distributed endpoint routing: the sgi spider chip," in *Symposium on High Performance Interconnects (Hot Interconnects)*, 1996, pp. 141–146.

[13] P. Gratz, B. Grot, and S. W. Keckler, "Regional congestion awareness for load balance in networks-on-chip," in *HPCA-16*, 2008.

[14] M. Hayenga, D. Johnson, and M. H. Lipasti, "Pitfalls of orion-based simulation," in *WDDD-10*, 2010.

[15] J. Howard and et al., "A 48-core ia-32 message-passing processor with dvfs in 45nm cmos," in *ISSCC*, 2010.

[16] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. V. Zyuban, H. M. Jacobson, and P. Bose, "Microarchitectural techniques for power gating of execution units," in *ISLPED*, 2004.

[17] N. E. Jerger and L. S. Peh, *On-Chip Networks, Synthesis Lecture in Computer Architecture*. Morgan and Claypool Publishers, 2003.

[18] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi, "Orion 2.0: A fast and accurate noc power and area model for early-stage design space exploration," in *DATE*, 2009.

[19] J. Kim, J. Balfour, and W. Dally, "Flattened butterfly topology for on-chip networks," *MICRO-40*, 2007.

[20] H. Matsutani, M. Koibuchi, D. Ikebuchi, K. Usami, H. Nakamura, and H. Amano, "Performance, area, and power evaluations of ultrafine-grained run-time power-gating routers for cmps," in *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 2011.

[21] H. Matsutani, M. Koibuchi, H. Amano, and D. Wang, "Run-time power gating of on-chip routers using look-ahead routing," in *ASP-DAC*, 2008.

[22] D. Meisner, B. T. Gold, and T. F. Wenisch, "Powernap: eliminating server idle power," in *ASPLOS*, 2009.

[23] H. Patil, R. Cohn, M. Charney, R. Kapoor, A. Sun, and A. Karunanidhi, "Pinpointing Representative Portions of Large Intel Itanium Programs with Dynamic Instrumentation," in *MICRO-37*, 2004.

[24] L.-S. Peh and W. J. Dally, "A Delay Model and Speculative Architecture for Pipelined Routers," in *Proceedings of the 7th International Symposium on High-Performance Computer Architecture*, 2001.

[25] A. Samih, R. Wang, A. Krishna, C. Maciocco, C. Tai, and Y. Solihin, "Energy-efficient interconnect via router parking," in *HPCA-19*, 2013.

[26] 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 *ISCA-30*, 2003.

[27] M. B. Taylor, J. S. Kim, J. E. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffmann, P. Johnson, J.-W. Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, M. Frank, S. P. Amarasinghe, and A. Agarwal, "The raw microprocessor: A computational fabric for software circuits and general-purpose programs," *IEEE Micro*, 2002.

[28] M. Thottethodi, A. R. Lebeck, and S. S. Mukherjee, "Self-tuned congestion control for multiprocessor networks," in *HPCA-7*, 2001.

[29] S. Volos, C. Seiculescu, B. Grot, N. Pour, B. Falsafi, and G. De Micheli, "Ccnoc: Specializing on-chip interconnects for energy efficiency in cache-coherent servers," in *NOCS-6*, 2012.

[30] H. Wang, L.-S. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," in *MICRO*, 2003.

[31] ——, "A power model for routers: Modeling alpha 21364 and infiniband routers," *IEEE Micro*, 2003.

[32] D. Wentzlaff, P. Griffin, H. Hoffmann, L. Bao, B. Edwards, C. Ramey, M. Mattina, C.-C. Miao, J. F. B. III, and A. Agarwal, "On-chip interconnection architecture of the tile processor," *IEEE Micro*, 2007.