TooT: An Efficient and Scalable Power-Gating Method for NoC Routers

Hossein Farrokhbakht, Mohammadkazem Taram, Behnam Khaleghi, and Shaahin Hessabi
Department of Computer Engineering, Sharif University of Technology, Tehran
Email: {hfarrokhbakht, taram, behnam_khaleghi}@ce.sharif.edu, hessabi@sharif.edu

***Abstract*—With the advent in technology and shrinking the transistor size down to nano scale, static power may become the dominant power component in *Networks-on-Chip* (NoCs). Power-gating is an efficient technique to reduce the static power of under-utilized resources in different types of circuits. For NoC, routers are promising candidates for power gating, since they present high idle time. However, routers in a NoC are not usually idle for long consecutive cycles due to distribution of resources in NoC and its communication-based nature, even in low network utilizations. Therefore, power-gating loses its efficiency due to performance and power overhead of the packets that encounter powered-off routers. In this paper, we propose *Turn-on on Turn* (TooT) which reduces the number of wake-ups by leveraging the characteristics of deterministic routing algorithms and mesh topology. In the proposed method, we avoid powering a router on when it forwards a *straight* packet or ejects a packet, i.e., a router is powered on only when either a packet turns through it or its associated node injects a packet. Experimental results on PARSEC benchmarks demonstrate that, compared with the conventional power-gating, the proposed method improves static power and performance by 57.9% and 35.3%, respectively, at the cost of a negligible area overhead.**

***Index Terms*—Power Gating, Network-on-Chip, Energy Consumption, Idle Time**

I. Introduction

The advent of multi-core processors with the ever increasing number of cores emphasizes the need for scalable, fast and energy efficient interconnects. However, existing scalable *Networks-on-Chip* (NoCs), e.g. 2D meshes, are critical consumers of the chip's total power budget [1], [2]. As an example, Intel Teraflop's on-chip network consumes up to 28% of the chip overall power [2]. Recent studies show that a significant portion of the NoC power consumption arises from routers' static power consumption [3], [4]. In addition, static power is predicted to be exacerbated by continuous scaling of transistor feature size. On the other hand, since the routers are provisioned for peak loads but usually operate at low average utilization levels [5], routers' static power are often viewed as an attractive target for power reduction. Nevertheless, this power reduction must be achieved without trading-off performance and scalability of NoC, which are also of vital importance because NoC should be able to meet the performance demands of future many-core processors.

There are several techniques for reducing static power in digital circuits [6]. Power gating is a widely used technique where idle blocks are powered-off for reducing static power [7]. Although this method can effectively reduce the power consumption in many digital circuits, it has several shortcomings particularly when applied to NoC routers. First, when a packet encounters a powered-off router, it has to wait until the router wakes up. Hence, power gating may impose a significant latency overhead to the NoC. The problem gets worse if the packet encounters further powered-off routers along its path. Second, although the purpose of power gating is to save static power when blocks are idle, power gating itself incurs power cost. In other words, powering on a router leads to a non-negligible power overhead. Thus, consecutive idle periods should be large enough to be able to compensate the imposed wake-up power overhead. While real applications have a relatively low average traffic load, even in these low utilization rates it is hard to find/predict large enough consecutive idle cycles as packets arrive intermittently and do not follow a particular arrival pattern [8], [9].

Several attempts have been made recently to overcome the power gating issues. A common optimization technique to alleviate the latency overhead of power gating is the early wake-up technique. In this technique, wake-up signals are generated multiple hops earlier before a packet arrives at the powered-off router. Thus, it can partially [10] or almost completely [11] hide the wake-up latency. However, in addition to its area overhead, this technique only addresses the wake-up latency issue associated with power gating, and still suffers from intermittent packet arrival. Other recent efforts in effective power-gating methods for routers, including [3], [12], [10], are either complex and impose significant area and performance overhead, or non-scalable. As an example, NoRD [3] avoids waking up a powered-off router using a ring to maintain connectivity of the network. Nevertheless, the long bypass ring makes the NoRD non-scalable, inasmuch as a packet may traverse a ring with the size of the network to evade encountering powered-off routers.

In this paper, we propose *Turn-on on Turn*, TooT, a scalable yet effective power gating method. TooT is based on a key observation, which will be justified later in this paper: most of the times a powered-off router needs to be woken-up is caused by arriving a *straight* packet. We define straight packets as the packets that a powered-off router receives from an adjacent router and do not have a turn on this powered-off router. For example, south to north and west to east packets are considered as straight packets. Based on this observation, we aim to devise a mechanism that minimizes the number

978-1-4673-9030-9/16/$31.00

of wake-ups by providing a bypass path for straight packets. In other words, we do not power on a router if the incoming packet does not turn through the powered-off router. Therefore, we efficiently mitigate both the latency and energy overheads of the power-gating technique. Based on our new design, we also introduce a new predictor for efficient detection of long idle periods to further enhance power consumption.

Simulation results using PARSEC benchmarks [13] show that: (1) TooT outperforms static power savings of the conventional power-gating by 63.6%, (2) Compared to the conventional power gating, TooT improves performance by 35.3%, and, (3) besides its simplicity (in terms of circuit overhead), TooT's predictor improves power-efficiency by 5.5% compared with its conventional counterparts.

The rest of the paper is structured as follows. Sec. II provides a condensed background on power gating, and motivates the need for a better approach. In Sec. III, we present TooT, our novel power-gating mechanism. Sec. IV discusses our evaluation methodology. Sec. V presents simulation results. Sec. VI reviews related work, and finally Sec. VII concludes the paper.

II. Background and Motivation

A. Requisiteness of Power Gating

Due to shrinking of transistor feature size and reduction in supply voltage, the contribution of leakage power in the chip's total power has been increased. On-chip networks, as a promising substitute for the old bus structures in today's *Multi-processor Systems-on-Chip* (MPSoCs), are not exceptions to this trend. While NoC by itself may consume up to 35% of a chip's total power [14], it has been shown that the ratio of static power in a typical 8×8 NoC increases from 43% in 45nm technology to more than 63% in 22nm [15], making the NoC's static power a major contributor to chip's power. Therefore, developing methods to alleviate the role of static power is indispensable.

Power gating is a widely accepted approach to eliminate the leakage power of unutilized/idle components in digital circuits. In order to efficiently apply power gating to a digital circuit, it requires idle time of the circuit to be long enough to compensate the corresponding wake-up energy overhead. While NoCs are adjusted to tolerate high inter-node communications to avoid performance bottleneck, real-world applications typically exhibit quiet traffics, leading to underutilized on-chip resources. Our analysis on PARSEC benchmarks on an 8×8 mesh topology reveals that, on average, 92.7% of routers are idle during application runtime. Altogether, first, power gating is crucial for NoCs due to the contribution of its leakage power to the total chip power, and second, it seems promising because of abounding idleness in components.

B. Coarse-Grained Power Gating of On-Chip Routers

Power gating is carried out by placing a switch (e.g. high threshold transistor(s)) between a circuit and its supply voltage. For on-chip routers, the switch is cut-off when all the data-paths, i.e. input buffers, output latches and crossbar

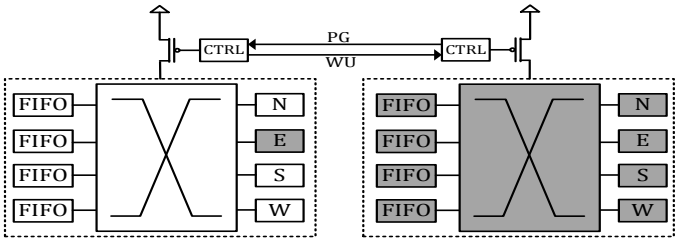


Fig. 1: Power gating of on-chip routers.

of a router are idle. It can be accomplished by a simple always-on monitoring controller. However, the adjacent routers must be notified of a power-gated router, otherwise they may send packets that will be thrown away in the network. Thus, handshaking signals are necessary to inform the neighbors if a router is powered-off. In this case, the neighbor tags the corresponding output port as powered-off, and make it unavailable when allocating the switch. Similarly, the adjacent routers need handshaking signals to inform a powered-off router when they send a packet for it. As illustrated in Fig. 1, the right side router is power gated and notifies its neighbors (only the left router is shown) by asserting the PG signals. Consequently, the neighbors tag the corresponding output buffer as powered-off (port E of the left router is tagged as powered-off). The neighboring routers prompt this router to power on via the wake-up (WU) signals.

C. Challenges of Power Gating the On-chip Routers

Cumulative wake-up latency: A router can remain in power-gated state until it confronts a packet. This happens when either a) its associated node injects a packet, b) the associated node is the destination of a packet (i.e., ejects a packet), or c) it transmits a packet to any neighbors. Each of the mentioned scenarios causes the router to wake up. Typically, waking up a router takes about 4ns [10], which corresponds to 8 cycles wake-up latency in 2 GHz frequency. The problem is exacerbated when a packet encounters several power-gated routers within its path. In conventional power gating method, when the network utilization is low, the majority of the routers are power gated, hence there is high probability that such scenario would happen, referred to as cumulative wake-up latency.

Break-even time: Each time a router is woken up, it imposes a considerable wake-up energy. The term *Break-Even Time* (BET) is defined as the minimum number of consecutive cycles that a gated block (here a router) must remain powered-off in order to compensate the wake-up energy overhead. For on-chip routers, the BET is about 10 cycles [10]. Therefore, power gating the idle intervals for less than 10 cycles imposes energy overhead rather than save. As mentioned before, we observed that routers are idle for 92.7% of time; however, 63.4% of idle intervals violate the BET role, i.e. are shorter than 10 cycles, that impose significant energy overhead and deteriorate the efficiency of power gating.

Based on the above-mentioned issues, applying efficient power gating on on-chip routers requires an approach to reduce the wake-up latency and energy in circuit level and/or reduce

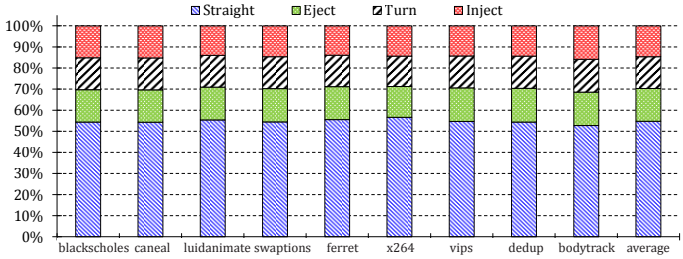


Fig. 2: Breakdown of packets encountering a powered-off router.

the number of wake-ups, which subsequently increases the idle intervals as well. While applying power gating in smaller granularities (e.g. buffer and port level) [16] may alleviate the BET issue, these methods are not efficient because of their hardware overhead and complexity. In addition, these methods do not mitigate the wake-up energy overhead and just split it to smaller pieces.

D. Key Observation

To further analyze the wake-ups of routers, we investigated the type(s) of packets that cause a powered-off router to wake-up in a commonly used mesh network with XY routing. Note that due to lower overhead of XY routing algorithm (compared with adaptive routing), it has become the dominant routing algorithm which is widely used in most commercial and research chips (e.g., [17], [18], [1]). In addition, the adaptive routing algorithms are usually beneficial for high loaded networks, but power-gating is usually used for low to medium traffic load, where there is little distinguishable difference between XY and adaptive routing algorithms [11]. Demonstrated in Fig. 2, straight packets are the majority of packets that interrupt power-gated routers, which on average contribute to 54.7% of wake-ups. The remaining part is distributed almost evenly between other types of packets, i.e., turn, inject and eject. Notice that the routing algorithm plays a key role here, since in the XY routing algorithm any packet is injected and ejected once, has at most one turn during its path, and mostly traverses straightly. As an obvious consequence, if we could evade waking-up the power-gated routers, performance overhead due to wake-up latency would be decreased significantly. In addition, not only will the wake-up energy overhead be mitigated, but also further energy can be saved since router will be in power-gated state instead of being in power-consuming *idle detection* state if conventional power gating method had been applied.

III. Proposed Method: TooT

A. Main Idea

In the conventional power-gating method, a powered-off router is woken up if any packet encounters the router. On the other hand, as noticed in Sec. II-D, more than 54.7% of these packets are straight packets. Accordingly, if we avoid powering on the routers for this large portion of packets, the efficiency of power-gating will highly improve by a) improving the power efficiency by increasing the length of intervals a router

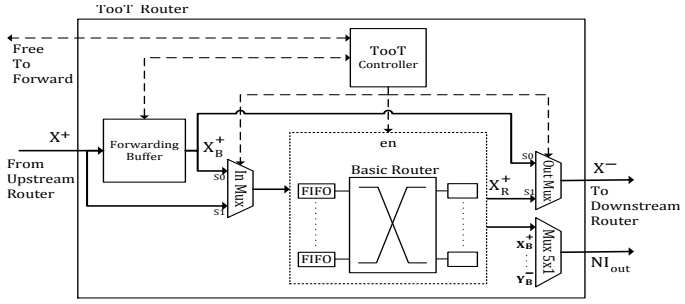


Fig. 3: TooT router architecture.

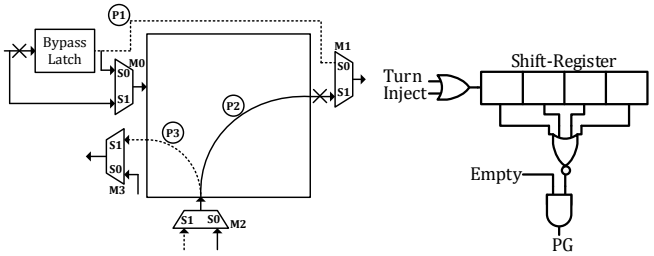
is powered-off, which not only increases the powered-off periods, but also reduces the energy-overheads incurred while powering-on a router as well, and b) reducing the accumulated wake-up latency by reducing the cases a packet waits for a power-gated router to be powered-on during its path from the source to the destination.

We prevent powering on the power-gated routers by augmenting all the routers with a bypass path. In addition, by appropriately architecting the bypass route, we aim to exploit it for the ejecting packets, as well (details of the bypass will be described in Sec. III-B). Thus, it is not anymore necessary to involve a power-gated router datapath to route such packets. Note that the observation of Fig. 2 is principal in the proposed method because otherwise, i.e. if all the packet types were distributed evenly, its effectiveness would be diminished, especially taking the potential overheads into account.

B. TooT Architecture

Fig. 3 demonstrates the proposed architecture to employ TooT. For the sake of simplicity, only the X^+ (the input from the west upstream router) to X^- (the output to the east downstream router) path is represented in this figure. For every bypass path such as X^+ to X^-, one bypass latch and two 2x1 multiplexers −all having equal flit widths− are needed. Once a flit arrives at a TooT router, depending on active/power-gated state of the router, different scenarios happen, as detailed in what follows.

Router is power-gated: Input flit enters the dedicated bypass latch. Afterwards, the controller checks whether the flit turns at the current router. To do that, it compares the destination coordination of the flit with that of the current router. If the flit is supposed to turn at this router, it must wait until the router wakes up (power-on signal is triggered by the controller). Otherwise, the flit is routed to the downstream router using the *S0* port of the *Out Mux*. Its selection signal is also controlled by the controller (represented by dashes in the figure). Note that for the output network interface, the output of the latch (denoted by $X_B{}^+$) passes through the *5x1 Mux*. Actually, in the power-gated mode, TooT router acts as a conventional router with one-flit input buffers, capable of merely forwarding straight packets and ejecting. Hence, it uses the *free-to-forward* signals to notify neighbors whether the corresponding bypass latches are empty. Similarly, if the



(a) (b)

Fig. 4: (a) Powered-off to powered-on transition. (b) TooT's predictor.

input buffers of the next router (or the bypass latch if it is power-gated) are not empty, the flit remains in bypass latch.

To avoid any conflict or starvation, while the bypass latch is not empty, the controller always grants the priority to this latch, i.e. *S0* port of *Out Mux* is passed to output, whether the router is power-gated or even active. The latter may happen when one or more flits inside the bypass latches cannot route to the next hops (due to network/router congestion, etc.) and meanwhile the router is triggered by some turning/injecting packets and has been powered-on (or is powering on). In such cases, the controller disables the (input ports of the) bypass latches until the router is completely woken-up, and then grants the priority to flit(s) that may reside in them. Finally, it returns the privilege to the basic router by selecting the port *S1* of the dedicated multiplexers. This procedure is depicted in Fig. 4a, where the router has been powered-on but there is a flit in the shown bypass latch. Both the flits in bypass latch and multiplexer $M2$ (or its corresponding input buffer inside the router) are about to send a flit through the port in $M1$. In this case, as mentioned above, priority is given to the latch (path is denoted by $P1$). However, simultaneous with $P1$, the *non-straight* path $P3$ also can be established since there is no full bypass latch (East to West) that sends a flit through $M3$. Since the router is powered-on (or is powering on) the input of the bypass latch is disabled.

Router is active: This scenario is similar to typical behavior of a router with the delay overhead of *In Mux* and *Out Mux* if the network frequency is the bottleneck. It is worth noting that as discussed above, when the router is active, all input flits enter the input buffers (through *S1* Port of *In Mux*). However, when there is a flit in a bypass latch, its corresponding TooT router output will be assigned to this latch. For instance, as depicted in Fig. 3, if the bypass latch of X^+ is full, the ${X_R}^+$ must wait until the controller grants the $S1$.

C. TooT's Predictor

In conventional power-gating methods, if a router has been idle for 4 consecutive cycles, it is deduced that it will be idle for the next 10 cycles, as well, providing it with the power-gating opportunity. However, since in the proposed method we have modified the wake-up conditions, it necessitates a change in predictor mechanism since it is not anymore necessary to keep a router on when the passing packets are straight/eject. Thus, we aim to power gate a router if no turning or injecting flit encounters it within 4 consecutive cycles. Therefore, as a key advantage of TooT, if frequent/bursty straight packets arrive at a router, the controller can simply powergate it after 4 cycles, which significantly improves the TooT efficiency. This is a substantial benefit to powergate a router during its operation because in on-chip networks, the traffic on a router's port is usually continuous. It is noteworthy that this 4-cycle predictor can be implemented with a simple circuitry. As shown in Fig. 4b, we have implemented the predictor with a (4-bit) shift register and a few logic gates. Based on this figure, in every cycle, if any of the flits traversing through the router is turn or inject, a logical 1 (one) is input into the shift register, which disables the power gating signal ($PG = 0$) for the next 4 cycles. On the other hand, if none of the outgoing flits is turn or inject, a 0 (zero) is shifted in. Repeating this procedure for 4 consecutive cycles makes the output of NOR gate logical 1. Then, if all of the router buffers, pipeline, etc. are empty ($Empty = 1$), the power-gating signal will be triggered.

IV. Evaluation Methodology

Simulation experiments are performed using *gem5* [19] full-system simulator, with the Ruby memory model and a modified version of *BookSim* [20]. The benchmark applications are taken from PARSEC benchmark suite [13]. Gem5 executed each application within their region of interests (ROI). Additionally, we integrated *DSENT* [4] with Booksim to model static and dynamic power for a 45 nm process. we used *Synopsys Design Compiler* with a *Nangate* 45 nm Open Cell Library to obtain area overhead of TooT. Our baseline platform is a 64-core processor with a 2-level cache hierarchy. Each core has a private L1 cache, while L2 cache is a distributed cache which is maintained via a MESI directory cache coherence protocol. We used an 8×8 2D mesh, with each router attached to a single processor core. Wakeup latency is set to 8 cycles assuming a 4ns wake-up delay [10] and 2GHz frequency. The BET is set to 10 cycles based on the evaluations in [10]. We also evaluated TooT across the full range of the network utilizations using synthetic workloads (*uniform*, *bit-complement*, and *shuffle* [20]). To this end, we warmed-up the simulator for 30,000 cycles and then collected network statistics for one milion cycles. For synthetic workloads, we used a mix of short 1-flit and long 5-flit packets. In order to assess the scalability of TooT, we also evaluate TooT under 4×4 and 16×16 network sizes. Table I summarizes the key parameters of the simulation setup.

TABLE I: Key Simulation Parameters

# of cores	64 on-chip, in-order, Alpha ISA, 2GHz
Private I/D L1$	32KB, 4-way, LRU, 2-cycle latency, MESI coherence protocol
Shared L2 per bank	8MB, 8-way, LRU, 8-cycle latency
Cache block size	64B
Memory	128 cycle access time, 4 on-chip memory controllers
Link bandwidth	128 bits/cycle
Network topology	4x4, 8x8, 16x16 mesh
Routing algorithm	XY
Router	3-stage and 4-stage
Virtual channel	1 VCs/VN, 3 VNs, 4 flits/VC

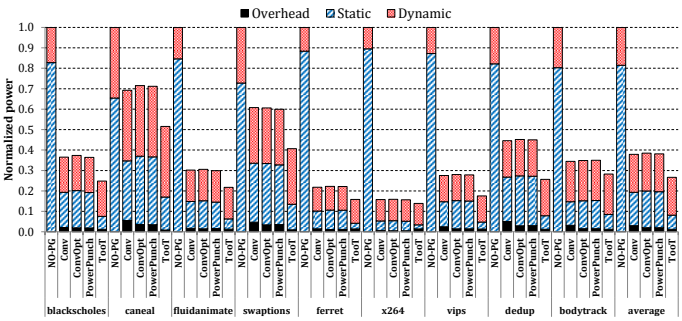


Fig. 5: Router power breakdown.

We compare the proposed method, TooT, with the following mechanisms:

- **NO-PG**: The baseline of our evaluations in which the power-gating technique is not applied to the routers.
- **Conv**: Conventional power-gating method which powers-off a router as soon as it goes to idle mode.
- **ConvOpt**: Like Conv but uses the early wake-up technique [10] to partially hide wake-up latency. Further, this method waits for 4 consecutive idle cycles before powering-off an idle router.
- **PowerPunch Signal**: A state-of-the-art power-gating method proposed by Chen et al. [11] which tries to completely hide wake-up latency.

V. Results and Analysis

In this section, we present the impact of TooT on power, energy and performance. Then, we analyze the TooT's behavior across the full range of network loads. Afterwards, we present the results of the scalability and sensitivity analysis of the proposed method with respect to the number of pipeline stages and wake-up latency. Finally, we report the area overhead of the TooT controller and datapath.

A. Impact on Power and Energy

Fig. 5 shows the breakdown of router power for the real workloads. The power values are normalized to NO-PG case. The total router power consumption is composed of dynamic power, static power, and power gating overhead. All the power wasted for applying power gating, such as the power consumed for powering on the routers, and the power-gating controller are accounted for the power-gating overhead.

In order to fairly compare the net savings in static power, in addition to static power itself, the power-gating overhead should also be considered. That is, the total of the bottom two bars in Fig. 5 shows the net static power. As it can be seen in Fig. 5, TooT reduces router static power consumption by 90% on average, while PowerPunch saves 75.9% and ConvOpt saves 75.5% of static power compared to NO-PG. TooT's considerable static power improvement is mainly achieved by the increased length of idle periods through bypass path, which has two consequences: 1) static power reduction through higher opportunity of powering off the routers, and 2) decreased power-gating overhead due to reduced number of wake-ups and violations of the BET role.

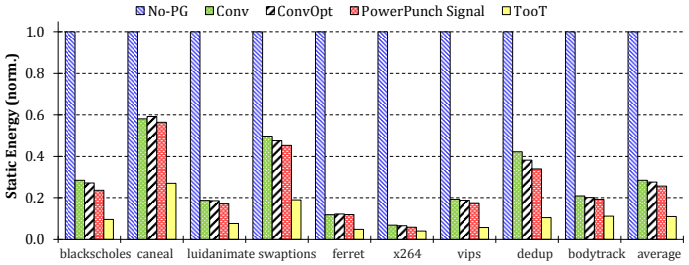


Fig. 6: Static energy comparison.

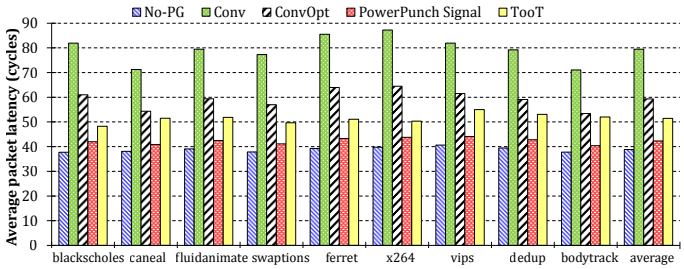


Fig. 7: Packet latency comparison.

Fig. 6 shows the static energy of different power-gating schemes normalized to NO-PG case. The figure shows that on average, TooT achieves 88.9% static energy saving, while PowerPunch and ConvOpt save 74.3% and 72.4% of static energy, respectively.

B. Impact on Performance

In addition to achieving a significant power saving over conventional power-gating methods, TooT is able to reduce packet latency, and hence improves performance. Fig. 7 shows the packet latency of the methods across the benchmarks. Results are normalized to the NO-PG case. It is noticeable that NO-PG leads to the least packet latency due to the fact that it does not power-off routers, and hence does not suffer from wake-up latency. On the other hand, Conv substantially increases average packet latency (104.4% on average). This is because the network utilization is low in these real workloads. Consequently, at any point in time a large number of routers are in powered-off mode. Therefore, packets should wait for many wake-ups, which leads to increase in average packet latency. Early wake-up technique which is applied in ConvOpt can partially hide the wake-up latency and reduce the large packet latency overhead incurred by power gating. However, this optimization technique cannot completely cover the wake-up latency and still leave non-negligible latency overhead (52.8% on average). TooT, on the other hand, on average imposes only 32.3% average packet latency overhead, and thus outperforms conventional power-gating methods. This improvement arises from the fact that TooT evades waking up powered-off routers for straight/eject packet. Thus, such packets do not have to wait for waking-up routers, which leads to a superior average packet latency compared to the conventional power-gating methods. PowerPunch is able to further reduce average packet latency, thanks to its signaling mechanism. However, as described in Sec. V-A, it is achieved at the cost of missing a significant power saving opportunity.

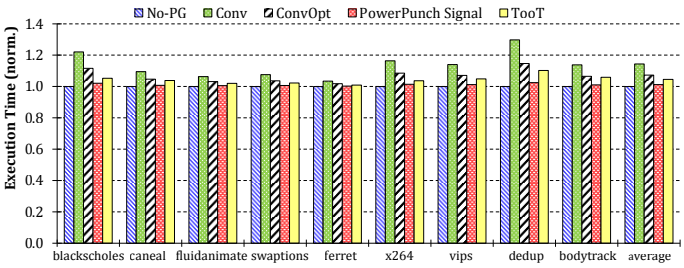


Fig. 8: Execution time comparison.

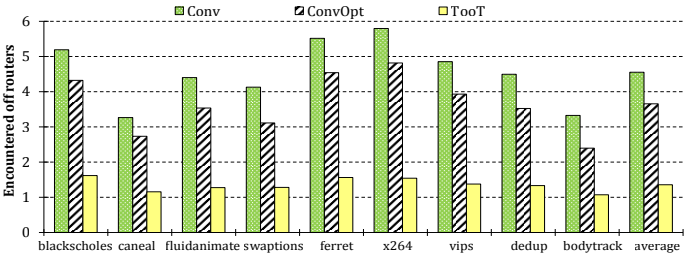


Fig. 9: Average number of encountered powered-off routers.

Execution time of benchmarks, as Fig. 8 depicts, follow a similar trend, although the network latency has different effects on different benchmarks. Some benchmarks, such as *dedup* and *blackscholes*, show more sensitivity to network latency than the others. Overall, the Conv, ConvOpt, PowerPunch and TooT increase the execution time by 14.4%, 7.3%, 1.2%, and 4.5% respectively, in order to reach the power saving described in Sec. V-A.

To give a better insight on how TooT can decrease the adverse effect of power gating on average packet latency, we present further analysis by providing the results of the average number of wake-ups a packet encounters across its path in network from the source to the destination. As it can be seen in Fig. 9, TooT reduces average number of wake-ups to 1.35, which is significantly less than the conventional power-gating schemes.

C. Impact of TooT Predictor

As discussed in Sec. III-C, we replace the conventional power-gating idle detection mechanism with a more effective predictor. In this section, we compare TooT's predictor with conventional power-gating idle detection mechanism. Fig. 10 represents the static power savings of TooT using different predictors. As the figure shows, TooT's predictor contributes to 5.5% of its static power improvement. TooT's predictor uses the straight packets as indicator of arrival of the future straight packets. Thus, it can power a router off earlier than conventional idle detection mechanism, which leads to more power saving.

D. Behavior across Full Range of Network Loads

Fig. 11 depicts the behavior of the power-gating schemes across different injection rates using synthetic workloads. Fig. 11 (a-c) compare the net static power (including static power and power gating overhead) of the schemes normalized to NO-PG case, and Fig. 11 (d-f) compare the average

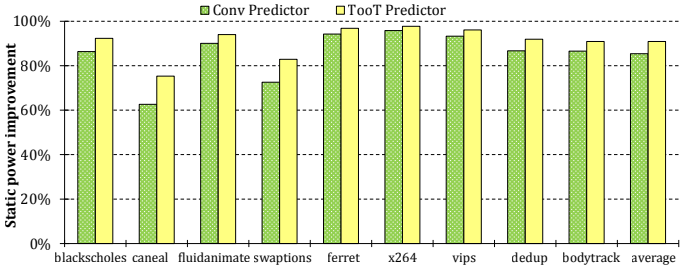


Fig. 10: TooT's predictor performance.

packet latency of the schemes. The figure shows that the net static power and average packet latency of all three synthetic workloads follow similar trends.

In low injection rates, where the power-gating schemes power off routers for majority of the time, conventional power-gating schemes can considerably reduce static power. Nevertheless, on occasions when a packet is injected into the network, it would encounter lots of wake-ups across its path to destination, which leads to a non-negligible average packet latency overhead. TooT, on the other hand, obviates the need for waking up a router when the packet is a straight packet, leading to a minor overhead and significant static power saving compared to related methods.

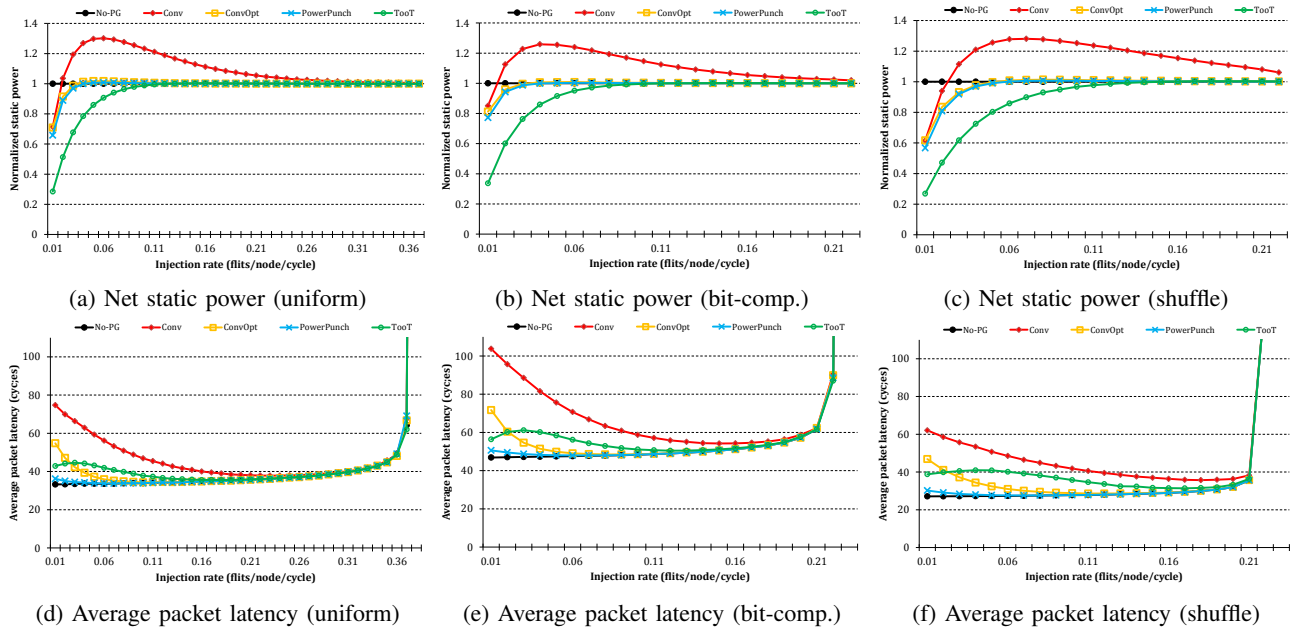
When injection rate gradually increases, first we observe a slight increase in TooT's average packet latency due to the congestion on TooT's bypass latch. Then, by further increasing the load, as routers incline to be in power-on state, packets tend to traverse through powered-on routers instead of bypass latch, which results in lower congestion in bypass latch, thereby reducing average packet latency. Note that although in some loads TooT may have slightly higher packet latency than ConvOpt, it achieves considerable higher static power saving in those injection rates. Besides, as our simulation shows, real application loads typically have very low injection rates.

As it can be observed, at some loads Conv even increases the net static power. This is because this method powers a router off as soon as it goes to idle mode (i.e., no packet in router's pipeline). Thus, in higher injection rates, where packets come usually at a router with an intervals of less than BET, the wake-up energy exceeds the static power savings of Conv; hence, this methods defeats the purpose of power gating.

Finally, in higher injection rates, where almost all routers are in powered-on state, net static power and average packet latency of the power-gating schemes conform to NO-PG case.

E. Sensivity Analysis on Pipeline Stages and Wake-up Latency

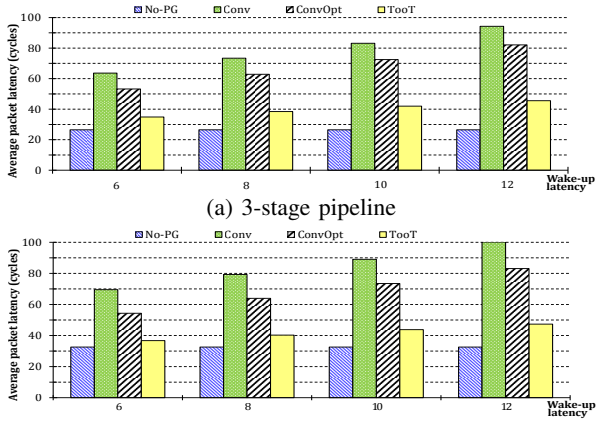
In order to investigate the effects of the number of pipeline stages and the wake-up latency on the average packet latency overhead of power-gating schemes, we set wake-up latency to 6, 8, 10, and 12. Then, we re-performed the simulations for routers with three and four pipeline stages. In these experiments we used uniform random workloads with an injection rate similar to the average injection rate of PARSEC benchmarks. Fig. 12 shows the results of these experiments. While the average packet latency of conventional power-gating methods substantially grows with increase of wake-up latency,



(a) Net static power (uniform) (b) Net static power (bit-comp.) (c) Net static power (shuffle)

(d) Average packet latency (uniform) (e) Average packet latency (bit-comp.) (f) Average packet latency (shuffle)

Fig. 11: Packet latency and net static power of synthetic workloads.



(a) 3-stage pipeline

(b) 4-stage pipeline

Fig. 12: Sensivity analysis on pipeline stages and wake-up latency.

TooT has considerably less average packet latency in all of the wake-up latencies.

F. Scalability and Area Overhead

Since TooT neither uses a centralized decision making mechanism nor a network-sized bypass ring, TooT is expected to be scalable. To examine the scalability of TooT, we performed simulations for 4×4 and 16×16 network sizes. In these experiments we used uniform random workloads with injection rate set to 0.01 flits/node/cycle. TooT outperforms the static power savings of ConvOpt by 60.3% and 50.2% for 16×16 and 4×4 networks, respectively. Also, TooT can reduce the average packet latency of ConvOpt by 24.2% and 11.8% for 16×16 and 4×4 networks, respectively. TooT's effectiveness (in terms of static power saving and average packet latency) is directly proportional to the size of the network. This is because in a mesh topology, a packet has at most one turn across its path. Thus, the ratio of straight packets is greater in larger network sizes, which leads to superior efficiency of TooT. We evaluated the area overhead of TooT using Synopsys Design Compiler and 45 nm technology node. Our experiments show that TooT's static power savings come at the cost of only 3.12% area overhead compared to conventional power-gating scheme.

VI. Related Work

Coarse-grained Power-Gating: *NoRD* [3] eliminates the dependency of a node to its associated router in order to provide it with the ability of injecting and ejecting packets without powering on the router. To this end, it adds an inport and outport bypass to *Network Interface* (NI) of the router, hence, any sent/received packet can inject/eject to/from the network without waking-up the router. Similarly, forwarded packets can bypass a powered-off router by entering through the dedicated inport bypass and passing through its outport bypass. While this solution eliminates the wake-up latency once a packet encounters a powered-off router, it increases the packet latency by detouring it to a neighbour router. The situation becomes worse when a packet faces multiple powered-off routers. Therefore, NoRD is not scalable for large NoC (e.g. an 8×8 mesh) because a packet may traverse long bypass rings −up to size of the network− to evade from powering on the router(s), which imposes significant packet latency for large NoC. In [11], in order to mitigate the wake-up latency, Chen et al. have proposed *PowerPunch* which aims to power on a powered-off router ahead arrival of the packets, using additional controlling signals. Since PowerPunch has nothing to do with the fragmented idleness period of routers and the number of wake-ups, this prevalent power-gating overhead is still an issue in this method. The authors in [12] propose *Router Parking* which powers off a subset −based on aggressive or conservative policies− of the routers that are connected to powered-off cores. It is also aware of maintaining

the network connectivity and limiting the latency imposed by the packets that detour the powered-off routers. Based on the information of the network traffic which is collected by a specific component, it chooses the subset of routers to be powered-off. It is only able to power-gate the routers connected to powered-off cores, so its efficiency is limited. In addition, Router Parking requires complex design because it updates the routing table once a router is powered-off, and modifies the routing algorithm. In [21] power-gating efficiency in Multi-NoC has been investigated. In a Multi-NoC, wires and buffers are split-off into several sub-networks wherein the node is connected to all of its associated routers in all the sub-networks. Thus, it is promising for power-gating since a sub-network can be power-gated entirely without affecting the network connectivity. To achieve this goal, they propose a subnet selection policy that provides it long idleness interval to alleviate the BET problem, and adapts the network bandwidth to application demands.

Fine-grained Power-Gating: *FlexiBuffer* has proposed partitioning a buffer into two unequal parts to facilitate the power-gating [22]. The smaller part is always active to provide routability in small traffics, while the second part is powered-off by default to save energy. When the number of flits in smaller part surpasses a determined threshold, the second part of the buffer is powered-on. In [16], Matsutani et al. have proposed ultra fine-grained power-gating for NoC routers in which each component (e.g. buffer, crossbar, etc.) can be individually powered-off, considering the network flow. In order to alleviate the wake-up latency, it uses 2-hop look-ahead signals to wake-up the buffers prior to arrival of the packets. While this method reduces the wake-up latency, it has no approach to reduce the number of wake-ups, i.e., idleness intervals of routers are still fragmented that leads to energy overhead because of wake-ups and also leaks power-gating opportunity due to idle-detection times. Finally in [23], *Panthre* is proposed to provide long sleeping intervals to fine-grained units to facilitate efficient power-gating. This method is based on the observation that only 10% of network traffic flows through the 30% of (low utilization) links, providing a potential for power-gating. Power-gating is done in link-level granularity, i.e., links with utilization lower than an adaptive threshold are powered-off. When any anomaly is detected, all links are woken-up, and for three consecutive successful epoch, the power-gating threshold reduces. In addition to complexity of added components, this method is inefficient since it can exploit only about half of its power-gating opportunity.

VII. Conclusion

NoC power consumption is considered as a significant and increasing portion of the overall multi-core chip power consumption. This work, Turn-on on Turn (TooT), focused on a novel power-gating mechanism in order to reduce the static power consumption of NoC routers. TooT, by providing a bypass path for straight and eject packets, avoids powering on a router when a straight/eject packet encounters a powered-off router. Consequently, TooT improves the efficiency of power-gating methods by reducing the overall number of wake-ups which leads to significant power and performance improvements. Full system simulations show that compared to an optimized conventional power-gating method, TooT is able to reduce static energy of NoC routers and average packet latency of network by 59.9% and 13.4%, respectively.

References

[1] J. Howard, S. Dighe *et al.*, "A 48-core ia-32 processor in 45 nm cmos using on-die message-passing and dvfs for performance and power scaling," *Solid-State Circuits, IEEE Journal of*, vol. 46, no. 1, pp. 173–183, 2011.

[2] Y. Hoskote, S. Vangal *et al.*, "A 5-ghz mesh interconnect for a teraflops processor," *Micro, IEEE*, vol. 27, no. 5, pp. 51–61, 2007.

[3] L. Chen and T. Pinkston, "Nord: Node-router decoupling for effective power-gating of on-chip routers," in *45th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2012, pp. 270–281.

[4] C. Sun, C.-H. O. Chen *et al.*, "Dsent-a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling," in *6th IEEE/ACM International Symposium on Networks on Chip (NoCS)*, 2012, pp. 201–210.

[5] R. Hesse, J. Nicholls, and N. D. E. Jerger, "Fine-grained bandwidth adaptivity in networks-on-chip using bidirectional channels," in *6th IEEE/ACM International Symposium on Networks on Chip (NoCS)*, 2012, pp. 132–141.

[6] N. Kim, T. Austin *et al.*, "Leakage current: Moore's law meets static power," *Computer*, vol. 36, no. 12, pp. 68–75, 2003.

[7] M. Keating, D. Flynn *et al.*, *Low Power Methodology Manual: For System-on-Chip Design*. Springer Publishing Company, Incorporated, 2007.

[8] R. Hesse and N. D. E. Jerger, "Improving DVFS in nocs with coherence prediction," in *9th ACM International Symposium on Networks on Chip (NoCS)*, 2015, pp. 24:1–24:8.

[9] M. Badr and N. D. E. Jerger, "Synfull: Synthetic traffic models capturing cache coherent behaviour," in *41th IEEE/ACM International Symposium onComputer Architecture (ISCA)*, 2014, pp. 109–120.

[10] H. Matsutani, M. Koibuchi *et al.*, "Run-time power gating of on-chip routers using look-ahead routing," in *13th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2008, pp. 55–60.

[11] L. Chen, D. Zhu *et al.*, "Power punch: Towards non-blocking power-gating of noc routers," in *21st IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2015, pp. 378–389.

[12] A. Samih, R. Wang *et al.*, "Energy-efficient interconnect via router parking," in *19th IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2013, pp. 508–519.

[13] C. Bienia, S. Kumar *et al.*, "The parsec benchmark suite: Characterization and architectural implications," in *17th ACM International Conference on Parallel Architectures and Compilation Techniques (PACT)*, 2008, pp. 72–81.

[14] J. S. Kim, M. B. Taylor *et al.*, "Energy characterization of a tiled architecture processor with on-chip networks," in *ACM International Symposium on Low Power Electronics and Design*, 2003, pp. 424–427.

[15] L. Chen, L. Zhao *et al.*, "Mp3: Minimizing performance penalty for power-gating of clos network-on-chip," in *20th IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2014, pp. 296–307.

[16] H. Matsutani, M. Koibuchi *et al.*, "Ultra fine-grained run-time power gating of on-chip routers for cmps," in *4th IEEE/ACM International Symposium on Networks on Chip (NoCS)*, 2010, pp. 61–68.

[17] P. Gratz, C. Kim *et al.*, "On-chip interconnection networks of the trips chip," *Micro, IEEE*, vol. 27, no. 5, pp. 41–50, 2007.

[18] B. K. Daya, C. H. O. Chen *et al.*, "Scorpio: A 36-core research chip demonstrating snoopy coherence on a scalable mesh noc with in-network ordering," in *41st IEEE/ACM International Symposium on Computer Architecture (ISCA)*, 2014, pp. 25–36.

[19] N. Binkert, B. Beckmann *et al.*, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, 2011.

[20] N. Jiang, D. Becker *et al.*, "A detailed and flexible cycle-accurate network-on-chip simulator," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2013, pp. 86–96.

[21] R. Das, S. Narayanasamy *et al.*, "Catnap: Energy proportional multiple network-on-chip," in *40th ACM International Symposium on Computer Architecture (ISCA)*, 2013, pp. 320–331.

[22] G. Kim, J. Kim, and S. Yoo, "Flexibuffer: Reducing leakage power in on-chip network routers," in *48th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2011, pp. 936–941.

[23] R. Parikh, R. Das, and V. Bertacco, "Power-aware nocs through routing and topology reconfiguration," in *51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2014, pp. 1–6.