Ultra Fine-Grained Run-Time Power Gating of On-Chip Routers for CMPs *

Hiroki Matsutani[1], Michihiro Koibuchi[2], Daisuke Ikebuchi[3], Kimiyoshi Usami[4],
Hiroshi Nakamura[1], and Hideharu Amano[3]

[1]The University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo, Japan
{matutani,nakamura}@hal.rcast.u-tokyo.ac.jp

[2]National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan
koibuchi@nii.ac.jp

[3]Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Japan
{ikebuchi,hunga}@am.ics.keio.ac.jp

[4]Shibaura Institute of Technology
3-7-5 Toyosu, Kohtoh-ku, Tokyo, Japan
usami@shibaura-it.ac.jp

Abstract

This paper proposes an ultra fine-grained run-time power gating of on-chip router, in which power supply to each router component (e.g., VC queue, crossbar MUX, and output latch) can be individually controlled in response to the applied workload. As only the router components which are just transferring a packet are activated, the leakage power of the on-chip network can be reduced to the near-optimal level. However, a certain amount of wakeup latency is required to activate the sleeping components, and the application performance will be degraded. In this paper, we estimate the wakeup latency for each component based on circuit simulations using a 65nm process. Then we propose four early wakeup methods to overcome the wakeup latency. The proposed router with the early wakeup methods is evaluated in terms of the application performance, area, and leakage power. As a result, it reduces the leakage power by 78.9%, at the expense of the 4.3% area and 4.0% performance when we assume a 1GHz operation.

1 Introduction

Recently, Network-on-Chips (NoCs) have been used in chip multi-processors (CMPs) to connect a number of processors and cache memories on a single chip, instead of traditional bus-based on-chip interconnects that suffer the poor scalability. Figure 1 illustrates an example of CMP inspired by [2], in which eight processors (or CPUs) and 64 L2 cache banks are interconnected by sixteen on-chip routers. These cache banks are shared by all processors and thus a cache coherence protocol is running on the CMP.

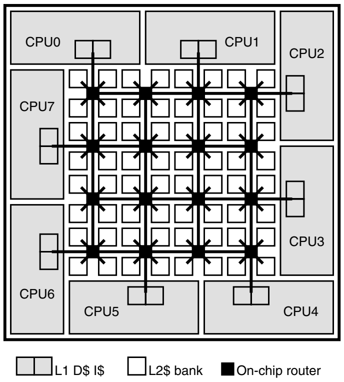


Figure 1. Example of 8-core CMP

NoCs can be evaluated from various aspects, such as the throughput, communication latency, hardware amount, and power consumption, but especially those used in CMPs are required to reduce the communication latency and power consumption. The communication latency is the primary performance factor, since it directly increases the cache access latency that affects the application performance on CMPs. As for the cost factor, the power consumption is becoming more and more important in almost all systems, since it affects the heat dissipation, packaging, and running costs of the system.

The power consumption is classified into dynamic switching power and static leakage power. The switching power is consumed only when packets are transferred on a NoC, while the leakage power (or static power) is consumed without any packet transfers as long as the NoC is powered on. Since the NoC is the communication infrastructure of CMPs, it must be always ready for the packet transfers at any workload so as not to increase the communication latency; thus a run-time power management that dynamically stops the leakage current whenever possible is highly required.

In this paper, we propose an ultra fine-grained run-time power gating of on-chip router, in which power supply to each router component (e.g., VC queue, crossbar MUX, and output latch) can be individually controlled in response to the

*This research was performed by the authors for STARC as part of the Japanese Ministry of Economy, Trade and Industry sponsored “Next-Generation Circuit Architecture Technical Development” program. The authors thank to VLSI Design and Education Center (VDEC) and Japan Science and Technology Agency (JST) CREST for their support. This work was also supported by Grant-in-Aid for JSPS Fellows.

applied workload. As only the router components which are just transferring a packet are activated, the leakage power of the on-chip network can be reduced to the near-optimal level. However, such techniques inherently increase the communication latency and degrade the application performance, since a certain amount of wakeup latency is required to activate the sleeping components. To mitigate the wakeup latency, we propose four early wakeup methods that can preliminarily detect the next packet arrival and activate the corresponding components. The proposed on-chip router with the early wakeup methods is evaluated in terms of the application performance, area, and leakage power.

The rest of this paper is organized as follows. Section 2 surveys coarse-grained and fine-grained power gating techniques. Section 3 designs the fine-grained power gating router and Section 4 proposes four early wakeup methods. Section 5 evaluates the power gating router with the early wakeup methods. Finally, Section 7 concludes this paper.

2 Power Gating Techniques

Power gating is a representative leakage-power reduction technique, which shuts off the power supply of idle circuit blocks by turning off (or on) the power switches which are inserted between the GND line and the blocks or between the VDD line and the blocks. This concept has been applied to circuit blocks with various granularities. Depending on the granularity of target circuit blocks (i.e., power domains), the power gating is classified into coarse-grained and fine-grained approaches.

Coarse-Grained Power Gating Each target circuit block is surrounded by a power/ground ring. Power switches are inserted between the core ring and power/ground IO cells. The power supply to the circuit block can be controlled by the power switches. Since the power supply to all cells inside the core ring is controlled at one time, this approach is well suited to the IP- or module-level power management. The coarse-grained approach has been popularly used, since its IP- or module-level power management is straightforward and easy to control. However, it typically imposes a microsecond order wakeup latency.

Fine-Grained Power Gating This approach has received a lot of attention in recent years because of its flexibility and short wakeup latency [6][13]. Although various types of fine-grained power gating techniques have been proposed, we focus on the method proposed in [13]. In this method, customized standard cells, each of which has a virtual ground (VGND) port by expanding the original cell, are used. These standard cells that share the same active signal form a single micro power domain, by connecting their VGND ports to a shared local VGND line, as shown in Figure 2. Power switches are inserted between the VGND line and GND line to control the power supply to the micro power domain. Figure 2 illustrates two micro power domains, each of which has its own local VGND line and power switch.

In this paper, we focus on the fine-grained approach because it is more suitable for on-chip routers than the coarse-grained one. Each input physical channel works independently of each other unless packet contentions with the other physical channels occur. In addition, all virtual channels in the same physical channel are not always used. Actually zero or only a few virtual channels are occupied at the same time in most cases. This indicates that a finer-grained partitioning can exploit the spatial and temporal communication locality and increase the power gating opportunities.

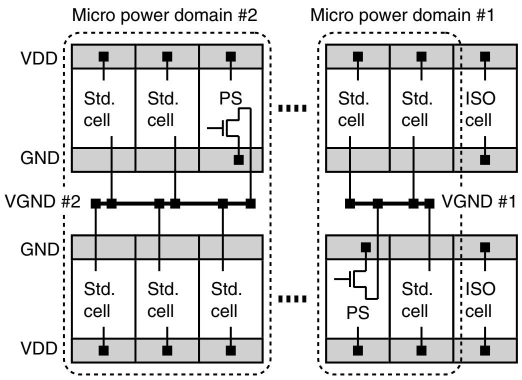


Figure 2. Concept of the fine-grained power gating. PS and ISO refer to a power switch and an isolation cell, respectively.

3 Ultra Fine-Grained Power Gating Router

Here, we propose an ultra fine-grained power gating router. We first show how an on-chip router is divided into a number of micro power domains. Then we implement these power domains using a 65nm process and evaluate them in terms of the area overhead and wakeup latency.

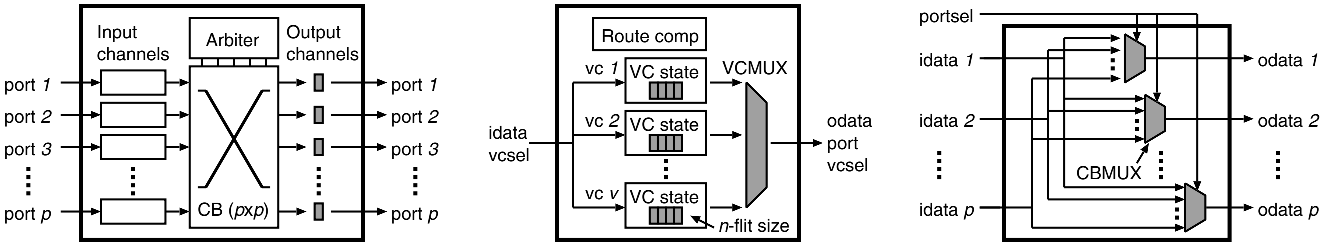
3.1 Power Domain Partitioning

Figure 3(a) illustrates a typical input-buffered wormhole router. The router has p input and output physical channels, a $p \times p$ crossbar switch, and a round-robin arbiter that allocates a pair of output virtual and physical channels for each incoming packet. Each input physical channel has a separated buffer queue for each virtual channel, while each output physical channel has a single 1-flit buffer or latch to decouple the switch and link delays.

Figure 3(b) shows an input physical channel that supports v virtual channels. It has a routing computation unit and a v-to-1 multiplexer (VCMUX) that selects only a single output from v virtual channels. Each virtual channel has a control logic, status registers, and an n-flit buffer queue.

Figure 3(c) shows a $p \times p$ crossbar switch. It is composed of p p-to-1 multiplexers (CBMUXes), each of which is controlled by a select signal from the arbiter.

Prior to partitioning the on-chip router into a number of micro power domains, we estimate the gate count of each router component, since the leakage power is proportional to the device area. An RTL model of the router is designed. Its parameters p, v, and n are set to 5, 4, and 4, respectively. The flit width w is set to 128-bit.



(a) Router overview (b) Input physical channel (c) Crossbar switch

Figure 3. Micro power domains of on-chip router. Each gray area denotes a power domain.

Table 1. Hardware amount of each router component (before PS insertion) [kilo gates]

Module	Count	Total gate count
4-flit VC buffer	20	111.06
1-flit output latch	5	5.49
5-to-1 CBMUX	5	4.91
4-to-1 VCMUX	5	4.21
Others	1	16.92
Total		142.58

Table 2. Hardware amount of each router component (after PS insertion) [kilo gates]

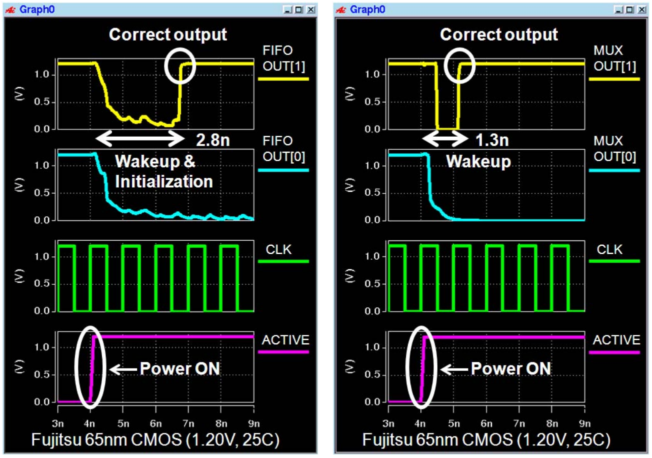
Module	Count	ISO	PS	Overhead
4-flit VC buffer	20	2.07	2.25	3.9% (15.4%)
1-flit output latch	5	0.51	0.16	12.2% (24.6%)
5-to-1 CBMUX	5	0.52	0.02	10.9% (23.3%)
4-to-1 VCMUX	5	0.54	0.02	13.3% (25.9%)
Others	1	0	0	0% (11.1%)
Total		3.64	2.44	4.3% (15.9%)

Table 1 shows the gate count of each router component. In this table, "Others" include the gate counts of routing computation units, an arbiter, VC status registers, and the other control logic, but each of these components is quite small compared to the other components (their total area is only 11.9% of the router). Thus, these miscellaneous logics are removed from the power domain list in order to simplify the power gating router design. Consequently, the router area is divided into 35 power domains including VC buffers, Output latches, VCMUXes, and CBMUXes, which can cover 88.1% of the total router area.

3.2 Power Domain Implementation

Here, we design all power domain types (i.e., VC buffer, Output latch, CBMUX, and VCMUX) in order to estimate their area overhead and wakeup latency.

The following design flow is used for all power domain types: 1) An RTL model of a power domain with an active signal is designed. 2) The RTL model is synthesized by the Synopsys Design Compiler. 3) Isolation cells are inserted to all output ports of the synthesized netlist in order to hold the output values of the domain when the power supply is stopped. 4) The netlist with isolation cells is placed by the Synopsys Astro. 5) The virtual ground (VGND) lines are formed, and the power switches are inserted between the VGND and GND lines by the Sequence Design CoolPower. 6) The netlist with power switches is routed by the Synopsys Astro. 7) The previous two steps are performed again in order to optimize the VGND, power switch sizing, and routing.



(a) 4-flit VC buffer domain (b) 5-to-1 CBMUX domain

Figure 4. Waveforms of circuit simulations

Using this design flow, we can obtain layout data (GDS files) of all power domain types. Notice this flow is fully automated; so an additional design complexity for the ultra fine-grained power gating is small.

Table 2 shows the area overhead of the isolation cells and power switches for each router component. In this table, ISO and PS show the total gate counts of isolation cells and power switches used in the router, respectively. In the Overhead column, a value without parentheses shows the area overhead of the isolation cells and power switches. The total area overhead of the power gating router is only 4.3%.

In this design flow, we used the customized standard cells each of which has a VGND port. We selected 106 cells from a commercial 65nm standard cell library and modified them to have a VGND port by expanding their cell height. In the Overhead column, a value with parentheses considers the area overhead of the customized standard cells against the original ones. In this case, the total area overhead is increased to 15.9% but it is still reasonable, since leakage current of these cells can be cut off when they are not activated.

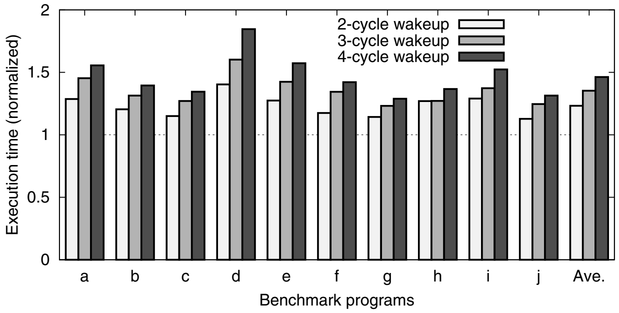


Figure 5. Execution time of SPLASH-2 benchmark without early wakeup methods.

3.3 Wakeup Latency Estimation

To estimate the wakeup latency of each power domain, the following steps are performed: 1) A spice netlist of the target power domain is extracted from the GDS file by the Cadence QRC Extraction. 2) The wakeup latency is measured based on circuit simulations of the spice netlist by the Synopsys HSIM.

Figure 4 shows the measured waveforms of the VC buffer and the CBMUX domains. The waveforms of the output latch and VCMUX domains are omitted, since they are quite similar to those of the VC buffer and CBMUX domains, respectively. In each figure, the first (top) and second waveforms show the lower two bits of the output (OUT[1] and OUT[0]). The third waveform shows the 1GHz clock signal and the fourth one shows the active signal. In these simulations, the lower two bits of input (IN[1] and IN[0]) are set to 1 and 0, respectively. Then, the active signal is asserted at the second rising edge of the clock. As a result, the output values of the VC buffer reach to the expected values within 2.8nsec, while those of the CBMUX take approximately 1.3nsec.

In this paper, we assume the wakeup latency of each power domain is two, three, and four cycles when the target NoC is operated at 667MHz, 1GHz, and 1.33GHz, respectively. This assumption is a little bit conservative, since the actual wakeup latencies are less than 3nsec as mentioned above.

3.4 Wakeup Latency Impact

NoCs on CMPs are quite latency sensitive, since certain communications (e.g., request and reply) are always required to access remote cache blocks. Thus, the wakeup latency of router components imposed by the power gating would degrade the application performance significantly.

To clearly show the negative impact of the wakeup latency, here we preliminarily evaluate the application performance on the CMP illustrated in Figure 1 without any early wakeup methods.

Figure 5 shows the execution cycles of SPLASH-2 benchmark that includes (a) radix, (b) lu, (c) fft, (d) barnes, (e) ocean, (f) raytrace, (g) volrend, (h) water-nsquared, (i) water-spatial, and (j) fmm. As wakeup latencies, two, three, and four cycles are assumed. Here, we omit the detailed simulation parameters (we will describe them in Section 5). In this graph, 1.0 indicates the execution time without power gating (i.e., 0-cycle wakeup). As shown, the average execution time of these applications is increased by 23.2%, 35.3%, and 46.3% when the wakeup latency is two, three, and four cycles, respectively.

Definitely, such performance penalty is unacceptable even if the leakage power is significantly reduced. To mitigate or remove the wakeup latency, an early wakeup method that can preliminarily detect the next packet arrival and activate the corresponding components is thus essential.

4 Wakeup Control Methods

This section proposes four wakeup control methods that mitigate the wakeup latency and improve the performance.

4.1 Naive Method

Naive method wakes up each micro power domain of a router as early as possible without any special modifications to the router. It assumes a conventional three-cycle router, in which a header flit is transferred through three pipeline stages that consist of the routing computation (RC) stage, the virtual channel and switch allocation (VSA) stage, and the switch traversal (ST) stage.

Assuming a packet is transferred from Router $(i-1)$ to Router i, each micro power domain of Router i is woken up as follows.

- **VC buffer:** Activation of an input VC buffer in Router i is triggered when a packet header in Router $(i-1)$ completes the VA and SA operations.
- **VCMUX:** Activation of a VCMUX in Router i is triggered when a packet header in Router $(i-1)$ completes the SA operation, since the VCMUX is shared by all virtual channels in the same input physical channel (see Figure 3(b)).
- **CBMUX:** Activation of all CBMUXes in Router i is triggered when a packet header in Router $(i-1)$ completes the SA operation.
- **Output latch:** Activation of all Output latches in Router i is triggered when a packet header in Router $(i-1)$ completes the SA operation.

In Naive method, every packet header must wait at Router $(i-1)$ until the activation of the VC buffer in Router i has been completed. This directly increases the communication latency and degrades the application performance.

Also, it seems to be inefficient that all CBMUXes and Output latches in Router i are woken up at the same time, although only a single pair of CBMUX and Output latch will be really used. However, routing computation in Router i must be completed before finding out which CBMUX and Output latch in Router i are really used. Since the activation requires multiple cycles (e.g., two cycles for 667MHz), it is difficult to wake up only the necessary CBMUX and Output latch without any additional latency penalty.

To mitigate or remove the above issues, the following sections propose more efficient wakeup methods.

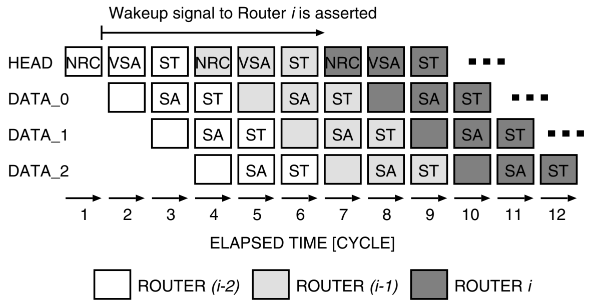


Figure 6. Router pipeline (Look-ahead)

4.2 Look-Ahead Method

Look-ahead method wakes up each micro power domain as early as possible by the look-ahead routing [5][12] that can detect which input channel of two hops away will be used. Since an activation of a micro power domain is triggered prior to several cycles before packets reach to the domain, it can mitigate or remove the negative impact of the wakeup latency.

Figure 6 illustrates how the look-ahead routing detects which input channel of two hops away will be used. In this figure, NRC denotes the routing computation for the next hop. Assuming a packet is transferred from Router $(i-2)$ to Router i via Router $(i-1)$, the NRC unit at Router $(i-2)$ computes the output channel of the next router (i.e., Router $(i-1)$), instead of its own output channel. Since the output channel of Router $(i-1)$ is directly connected to an input channel of Router i, the NRC unit at Router $(i-2)$ can detect which output channel of Router $(i-1)$ and which input channel of Router i will be used.

As shown in Figure 6, Router $(i-2)$ can trigger the activation of Router i when it completes its NRC. There is a five cycle margin after a packet completes the NRC of Router $(i-2)$ until the packet actually reaches to Router i.

Activation of each micro power domain using Look-ahead method is summarized as follows.

- **VC buffer:** Activation of an input VC buffer in Router i is triggered when a packet header in Router $(i-2)$ completes the NRC operation.
- **VCMUX:** Activation of a VCMUX in Router i is triggered when a packet header in Router $(i-2)$ completes the NRC operation.
- **CBMUX:** Activation of a CBMUX in Router i is triggered when a packet header in Router $(i-1)$ completes the NRC operation.
- **Output latch:** Activation of an Output latch in Router i is triggered when a packet header in Router $(i-1)$ completes the NRC operation.

Notice that Naive method wakes up all CBMUXes and Output latches at once without considering which output channel will be used, so as not to increase wakeup latency. Compared to Naive method, Look-ahead method is more efficient, since it wakes up only a single pair of CBMUX and Output latch which will be used in several cycles later.

An input channel (or NRC unit) of Router $(i-2)$ has to deliver the wakeup signal to the corresponding VC buffer and VCMUX of Router i. Also, the NRC unit of Router $(i-1)$ has to deliver the wakeup signal to the corresponding CBMUX and Output latch of Router i. To deliver these signals, a wakeup control network is needed. The wakeup signal spans the twice longer distance than a wire between two neighboring routers; thus an additional cycle would be required to deliver the wakeup signal, depending on the distance between two routers.

Another difficulty of Look-ahead method is the wakeup control of the first hop. We assume that the source network interface (source NI) can trigger the activation of the first and second hops during the packetization. However, assuming the source NI triggers the activation of the first hop one cycle ahead, it compensates only one cycle of the first-hop wakeup latency but suffers the remaining latency.

The i-th hop router can mitigate $T^i_{recover}$ cycles of the wakeup latency. $T^i_{recover}$ is calculated as follows.

$$T^i_{recover} = \begin{cases} 2n - T_{wire} - 1 & i \geq 2 \\ 1 & i = 1 \end{cases}, \qquad (1)$$

where n is the router pipeline depth (e.g., three stages) and T_{wire} is the wire delay of a wakeup signal. Assuming $n = 3$ and $T_{wire} = 1$, the second or farther hop routers can mitigate up to four cycles of the wakeup latency, though the first hop mitigates only a single cycle.

4.3 Look-Ahead with Ever-On Method

Look-ahead with ever-on method is an extension of the original Look-ahead method to mitigate the wakeup latency of the first hop. In the ever-on method, VC buffers which are activated frequently as the first hop are selected as "ever-on" domains, and their power supplies are never stopped. The ever-on domains must be selected quite carefully, since they always consume leakage power. The other power domains are woken up in the same way as the original Look-ahead.

To select the ever-on domains, we first analyzed traffic patterns of the target CMP illustrated in Figure 1. As a result, VC buffers which are directly connected from CPU cores are selected as ever-on domains in this paper. We will evaluate the impact of these ever-on domains in terms of the application performance and leakage power in Section 5.

4.4 Look-Ahead with Active Buffer Window (ABW) Method

Look-ahead with active buffer window (ABW) method is another extension of the original Look-ahead method to completely remove the first-hop wakeup latency at the expense of leakage power. This method is inspired by a leakage-aware buffer management proposed in [4].

In the ABW method, each VC buffer domain is divided into more finer flit-level power domains, each of which can be activated and deactivated independently. For example, a 4-flit VC buffer is divided into four flit-level power domains. To store incoming flits without any first-hop wakeup latency, a certain number of the flit-level power domains in each VC buffer are always kept active, regardless of the workload. The activated part of the VC buffer is called "active buffer window". The active buffer window size is always kept. That is,

Table 3. Simulation parameters (CMP)

Processor	UltraSPARC-III
L1 I-cache size	16 KB (line:64B)
L1 D-cache size	16 KB (line:64B)
# of processors	8
L1 cache latency	1 cycle
L2 cache size	256 KB (assoc:4)
# of L2 cache banks	64
L2 cache latency	6 cycle
Memory size	4 GB
Memory latency	160 cycle

Table 4. Simulation parameters (NoC)

Topology	4×4 mesh
Routing	dimension-order
Switching	wormhole
# of VCs	4
Buffer size	4 flit
Router pipeline	[RC][VSA][ST]
Flit size	128 bit
Control packet	1 flit
Data packet	5 flit

the active buffer window in a VC buffer moves whenever a part of the active buffer is consumed, in order to prepare for the next flit arrival. Assuming that the active buffer window size is two, two flit-level power domains are activated to prepare for the next flits. The other power domains are woken up in the same way as the original Look-ahead method.

5 Evaluations

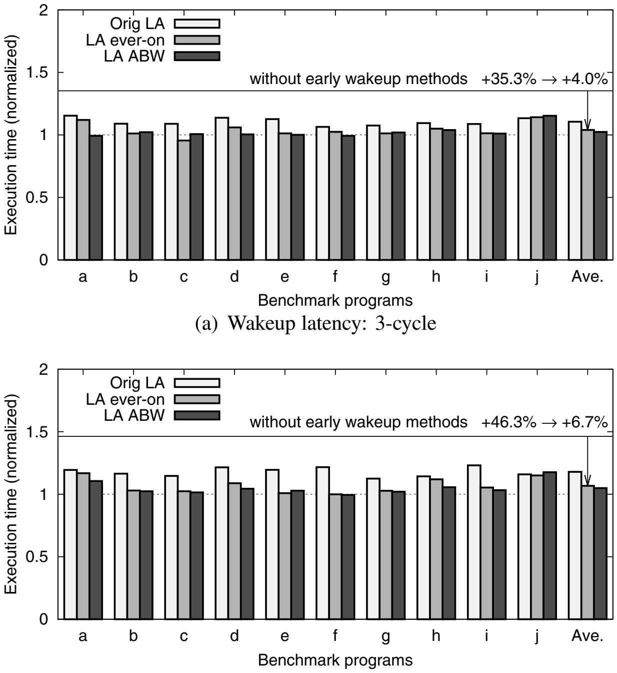
This section evaluates the ultra fine-grained power-gating router with the early wakeup methods in terms of the application performance and leakage power.

5.1 Simulation Environment

A NoC used in the 8-core CMP illustrated in Figure 1 is simulated. The cache architecture is SNUCA [7] and a cache coherence protocol is running on it. Table 3 lists the processors and memory system parameters. Table 4 shows the on-chip routers and network parameters. To simulate the above-mentioned CMP, we use a full-system multi-processor simulator: GEMS [10] and Virtutech Simics [8].

Network Model: We modified a detailed network model of GEMS, called Garnet [1], in order to accurately simulate the proposed ultra fine-grained power gating of on-chip routers and the four wakeup methods. As shown in Table 4, typical 3-stage pipelined routers are used in the NoC. In the Look-ahead based wakeup methods, the wire delay of a wakeup signal T_{wire} is set to one cycle. The size of a VC buffer in the router is set to four flits. In the ABW method, the active buffer window size is set to two flits.

Cache Coherence Protocol: Token coherence protocol [9] is used. To avoid end-to-end protocol (i.e., request-reply) deadlocks, the on-chip network uses four virtual channels (VC0 to VC3) as follows.



(a) Wakeup latency: 3-cycle

(b) Wakeup latency: 4-cycle

Figure 7. Execution time of SPLASH-2 benchmark with different wakeup methods.

- **VC0:** Request from L1$ to L2$ bank; Request from L2$ bank to L1$
- **VC1:** Request from L2$ bank to directory controller; Request from directory controller to L2$ bank
- **VC2:** Reply from L1$/directory to L2$ bank; Reply from L2$ bank to L1$/directory
- **VC3:** Persistent request from L1$

The utilization ratio of each virtual channel is different. For example, the utilization of VC1 is low when the main memory is accessed sparsely due to frequent cache hits. VC3 is assigned to the persistent requests for avoiding the starvation, but its traffic amount is quite small since such situation is not so frequent (e.g., 0.19% of all requests [9]). Our ultra fine-grained power gating technique can exploit such imbalanced use of power domains inside a router.

Benchmark Programs: To evaluate the application performance of the proposed fine-grained power gating with different wakeup methods, we use ten parallel programs of SPLASH-2 benchmark [15]. Sun Solaris 9 operating system is running on the 8-core CMP. These benchmark programs are compiled by Sun Studio 12 and are executed on Solaris 9. The number of threads is set to eight in each program.

5.2 Application Performance

Here, we count the execution cycles of the ten benchmark programs when the proposed ultra fine-grained power gating technique is applied to on-chip routers in the CMP. We also compare the proposed early wakeup methods in terms of the application performance. As derived in Section 3.3,

Table 5. Leakage power of each router component [uW]

Module	Count	Total leakage power
4-flit VC buffer	20	189.07
1-flit output latch	5	16.71
5-to-1 CBMUX	5	11.41
4-to-1 VCMUX	5	13.45
Others	1	38.36
Total		269

the wakeup latency of each power domain is less than 3nsec; thus we assume the wakeup latency is two, three, and four cycles in our simulations when the target NoC is operated at 667MHz, 1GHz, and 1.33GHz, respectively.

Figure 7(a) shows the application performance of the original Look-ahead, Look-ahead with ever-on, and Look-ahead with ABW methods [1], when the wakeup latency of every domain is set to three cycles. The benchmark set includes (a) radix, (b) lu, (c) fft, (d) barnes, (e) ocean, (f) raytrace, (g) volrend, (h) water-nsquared, (i) water-spatial, and (j) fmm. Their application performance is normalized so that the original application performance without power gating (i.e., 0-cycle wakeup) is 1.0. As shown in the graph, all the programs indicate a similar tendency. Although the power gating with no early wakeup methods increases the execution time by 35.3% (see Section 3.4), those with the original Look-ahead, Look-ahead ever-on, and Look-ahead ABW methods increase 10.5%, 4.0%, and 2.4% on average, respectively. Look-ahead ever-on and Look-ahead ABW methods can successfully mitigate the wakeup latency when the target NoC is running at 1GHz.

Figure 7(b) shows the application performance when the wakeup latency is four cycles assuming an operating frequency of 1.33GHz. The power gating with no early wakeup methods increases the execution time by 46.3%, while those with Look-ahead ever-on and ABW methods increase only 6.7% and 4.9%, respectively. Thus, these early wakeup methods are still reasonable at such a high operating frequency.

5.3 Leakage Power Reduction

In this section, we estimate the average leakage power of the proposed power gating router with different wakeup methods when the application workload is applied to it.

Table 5 shows leakage power of each router component, based on the post-layout design of the power-gating router implemented in Section 3.2. The run-time power gating is applied to VC buffer, Output latch, VCMUX, and CBMUX, while the others are not. We used the 106 customized standard cells based on a low-power version of a commercial 65nm standard cell library. Temperature and core voltage were set to 25C and 1.20V, respectively. These leakage parameters were fed to the full system CMP simulator, in order to evaluate the run-time leakage power of the routers when the application programs are running on them.

To clearly show the leakage power reduction of each power domain type, the proposed fine-grained power gating is gradually applied as the following three steps.

- **Level 1:** VC buffers are power gated.
- **Level 2:** VC buffers, VCMUXes, and CBMUXes are power gated.
- **Level 3:** VC buffers, VCMUXes, CBMUXes, and Output latches are power gated.

Figure 8(a) shows an average leakage power of the router when Level 1 power gating, which covers only VC buffers, is applied. The wakeup latency of all power domains is set to three cycles assuming an operating frequency of 1GHz. In this graph, 100% indicates the leakage power of the router without power gating (i.e., 269uW).

The original Look-ahead method shows the smallest leakage power in these methods, while it cannot avoid the first-hop wakeup latency and degrades the application performance, as shown in Section 5.2. Look-ahead with ABW method consumes the largest leakage power since an active buffer window (i.e., two flits) of each VC buffer is always activated, although it achieves the best performance. Look-ahead with ever-on method consumes a little bit more leakage power compared to the original Look-ahead method, since it has some ever-on power domains to mitigate the first-hop wakeup latency. Fortunately, the leakage power of these ever-on domains is not crucial, since they are limited to VC buffers of VC0 and VC2 in input physical channels directly connected from processor cores. As a result, Look-ahead with ever-on method is the best choice to balance the performance and leakage power. In Level 1 power gating, the ever-on method reduces the average router leakage power by 64.6%.

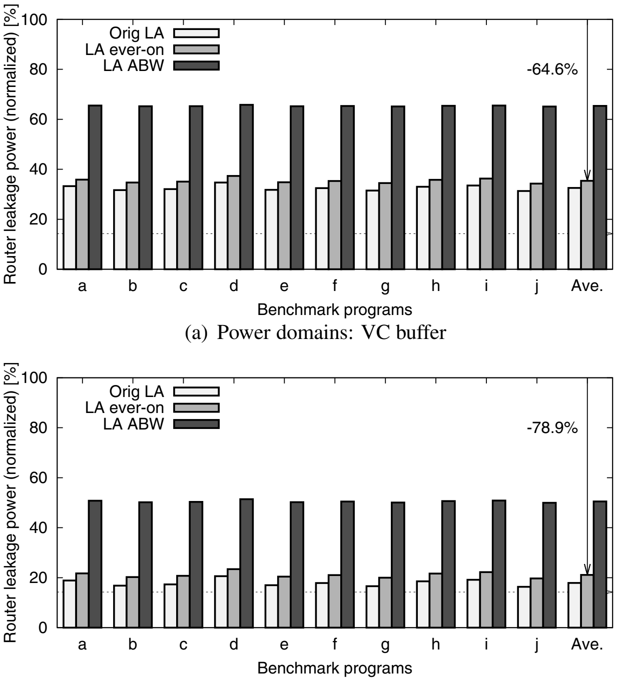
Figure 8(b) shows an average leakage power of the router when our ultra fine-grained power gating is fully applied. In this Level 3 power gating, Look-ahead with ever-on method reduces the average router leakage power by 78.9%.

6 Related Work

As the standby power consumption is becoming more and more serious, various power gating techniques have been applied to on-chip routers to reduce the standby leakage power [11][12][14]. In [14], each router is partitioned into 10 smaller sleep regions with control of individual router ports. An input physical channel level power gating is studied in [12], while a virtual channel level power management is discussed in [11]. In [3], PMOS power switches controlled by an ultra-cut-off (UCO) technique are inserted on each NoC unit to maintain minimum leakage in standby mode.

Compared to these approaches, we employ a more finer approach in which each router is partitioned into 35 micro power domains in order to fully exploit the spatial and temporal communication locality inside and outside a router. Existing input buffer power gating [11][12] is corresponding to our Level 1 power gating, while our Level 3 approach covers more area and reduces more leakage power, as shown in Section 5.3. Moreover, in this paper, we designed the proposed ultra fine-grained power gating router with a 65nm process in the same manner as in [6]. We also showed the actual area overhead and wakeup latency, based on this design.

[1] We omitted Naive method, since its performance is evidently inferior to the original Look-ahead method.



(a) Power domains: VC buffer

(b) Power domains: VC buffer, VCMUX, CBMUX, and Output latch

Figure 8. Average leakage power of an on-chip router with different wakeup methods.

When the power gating techniques are applied to on-chip networks, the wakeup control of power domains is one of the most important factors on the application performance. In [4], as a leakage-power aware buffer management method, a certain portion (i.e., window size) of the buffer is made active before it is accessed, in order to remove the performance penalty. Also, our original Look-ahead method is inspired by [12]. However, [12] considers only the wakeup control for input physical channels, while this paper proposes those for VC buffers, VCMUXes, CBMUXes, and Output latches, each of which is activated in different conditions.

7 Summary and Future Work

On-chip communications inherently have a strong spatial and temporal locality inside and outside a router. To fully exploit the locality, we proposed an ultra fine-grained power gating router, in which power supply to each router component (e.g., VC buffer, VCMUX, CBMUX, and Output latch) can be individually controlled in response to the workload. We implemented the power gating router that consists of 35 micro power domains using a 65nm process. The power gating router with the early wakeup methods was evaluated in terms of the area, application performance, and leakage power. The area overhead for the power switches and isolation cells is only 4.3%, while the customized cell height currently increases the overhead to 15.9%, although there is room to be optimized. The simulation results show that the router leakage power is reduced by 78.9%, even when application programs are running, at the expense of the 4.0% performance overhead when we assume a 1GHz operation.

As future work, we will focus on the following issues.

- More accurate trade-off analysis: Inserting power switches negatively affects the critical path delay. Also, a small amount of energy is consumed by power switches and a single-bit wakeup signal to wake up a power domain. More sophisticated wakeup policies that take into account the overhead energy are required.
- Routing protocols: Look-ahead wakeup methods are designed for deterministic routing. We will consider possibilities to apply them to oblivious and adaptive routings.
- Power gating for the network interfaces: We are considering the use of various information from the coherence protocol or operating system to guide the run-time power gating decisions.

References

[1] N. Agarwal, L.-S. Peh, and N. Jha. Garnet: A Detailed Interconnection Network Model inside a Full-system Simulation Framework. Technical Report CE-P08-001, Princeton University, 2008.

[2] B. M. Beckmann and D. A. Wood. Managing Wire Delay in Large Chip-Multiprocessor Caches. *Proceedings of the International Symposium on Microarchitecture (MICRO'04)*, pages 319–330, Dec. 2004.

[3] E. Beigne, F. Clermidy, H. Lhermet, S. Miermont, Y. Thonnart, X.-T. Tran, A. Valentian, D. Varreau, P. Vivet, X. Popon, and H. Lebreton. An Asynchronous Power Aware and Adaptive NoC Based Circuit. *IEEE Journal of Solid-State Circuits*, 44(4):1167–1177, Apr. 2009.

[4] X. Chen and L.-S. Peh. Leakage Power Modeling and Optimization in Interconnection Networks. *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED'03)* , pages 90–95, Aug. 2003.

[5] M. Galles. Spider: A High Speed Network Interconnect. *IEEE Micro*, 17(1):34–39, 1997.

[6] D. Ikebuchi et al. Geyser-1: A MIPS R3000 CPU Core with Fine Grain Runtime Power Gating. *Proceedings of the IEEE Asian Solid-State Circuits Conference (A-SSCC'09)*, pages 281–284, Nov. 2009.

[7] C. Kim, D. Burger, and S. W. Keckler. An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches. *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'02)*, pages 211–222, Oct. 2002.

[8] P. S. Magnusson et al. Simics: A Full System Simulation Platform. *IEEE Computer*, 35(2):50–58, Feb. 2002.

[9] M. M. K. Martin, M. D. Hill, and D. A. Wood. Token Coherence: Decoupling Performance and Correctness. *Proceedings of the International Symposium on Computer Architecture (ISCA'03)*, pages 182–193, June 2003.

[10] M. M. K. Martin, D. J. Sorin, B. M. Beckmann, M. R. Marty, M. Xu, A. R. Alameldeen, K. E. Moore, M. D. Hill, and D. A. Wood. Multifacet General Execution-driven Multiprocessor Simulator (GEMS) Toolset. *ACM SIGARCH Computer Architecture News (CAN'05)*, 33(4):92–99, Nov. 2005.

[11] H. Matsutani, M. Koibuchi, D. Wang, and H. Amano. Adding Slow-Silent Virtual Channels for Low-Power On-Chip Networks. *Proceedings of the International Symposium on Networks-on-Chip (NOCS'08)* , pages 23–32, Apr. 2008.

[12] H. Matsutani, M. Koibuchi, D. Wang, and H. Amano. Run-Time Power Gating of On-Chip Routers Using Look-Ahead Routing. *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC'08)* , pages 55–60, Jan. 2008.

[13] K. Usami and N. Ohkubo. A Design Approach for Fine-grained Run-Time Power Gating using Locally Extracted Sleep Signals. *Proceedings of the International Conference on Computer Design (ICCD'06)*, Oct. 2006.

[14] S. R. Vangal et al. An 80-Tile Sub-100-W TeraFLOPS Processor in 65-nm CMOS. *IEEE Journal of Solid-State Circuits*, 43(1):29–41, Jan. 2008.

[15] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta. SPLASH-2 Programs: Characterization and Methodological Considerations. *Proceedings of the International Symposium on Computer Architecture (ISCA'95)*, pages 24–36, June 1995.