Chapter 3

Duty Buffer Based Power Gating Approach

Peng Wang, Sobhan Niknam, Zhiying Wang, Todor Stefanov,
"A Novel Approach to Reduce Packet Latency Increase Caused by Power Gating in Network-on-Chip"
in Proceedings of the 11th IEEE/ACM International Symposium on Networks-on-Chip (NOCS), Seoul, South Korea, 2017, pp. 1–8.

In this chapter, we present in more detail our duty buffer based (DB-based) power gating approach, which corresponds to **Contribution 1** introduced in Section 1.4, to solve the research **Problem 1**, described in Section 1.3.1. This chapter is organized as follows. Section 3.1 further introduces the research problem of the packet latency increase caused by power gating. It is followed by Section 3.2, which gives a summary of the contributions in this chapter. Then, Section 3.3 gives an overview of the related work. Section 3.4 elaborates on our novel duty buffer (DB) structure and the DB-based power gating approach. Finally, Section 3.5 introduces the experimental setup and shows the results, and Section 3.6 concludes and discusses this chapter.

3.1 Problem Statement

As we have shown in Section 1.1.1, the NoC accounts for 10% to 36% of the total power consumption in many-core systems. Due to the low average traffic load of real applications [DMMD09], most of the NoC power consumption is contributed by the static power consumption of idle routers. Even, under a minimal resource configuration, the NoC router static power consumption is still about 64% [CZPP15] of the total NoC power consumption. As a consequence, it is critical to reduce the high static power consumption of the routers.

On the other hand, NoCs have the characteristics of a distributed structure, a naturally unbalanced traffic workload, and a low average injection traffic rate, which make power gating being an applicable and effective way of powering off idle NoC routers to reduce the power consumption. However, there is a notable wakeup delay to power on the powered-off components during the wakeup process. When power gating is applied on a NoC, this wakeup delay, about 6-12 clock cycles [CZPP15], will interrupt the close cooperation between routers and will block the routing path for a while. As a consequence, the packet latency over the whole routing path dramatically increases and the NoC performance is degraded.

Several works [ZOG+15, MKWA08] try to reduce the packet latency increase caused by the power gating. As the drowsy SRAM has only 2 clock cycles wakeup delay, Zhan [ZOG+15] uses the drowsy SRAM to build virtual channels (VCs) in a NoC router and implements a fine-grained power gating on virtual channels. In this way, the wakeup process becomes faster. By sending a wakeup signal ahead of the packet injection, Matsutani [MKWA08] switches on the power of the powered-off routers earlier, thereby hiding part of the wakeup delay. These approaches [ZOG+15, MKWA08] can reduce or hide part of the wakeup delay in a single wakeup process. But a packet may experience multiple wakeup processes and accumulate large delay along the routing path. In fact, as the average traffic load of real applications is low [DMMD09], there is high probability for packets to experience multiple wakeup processes. Furthermore, with more cores integrated on a chip in future many-core systems, this cumulative wakeup delay becomes much higher.

3.2 Contributions

In order to further reduce the packet latency increase caused by the power gating, in this chapter, we propose a novel and flexible hardware structure, called Duty Buffer (DB). Based on the DB structure, we propose a DB-based power gating approach to reduce the static power consumption of the VCs in routers. By using our DB to temporarily replace a powered-off VC and accept packets, an upstream router does not need to block packets while waiting for the VC in a downstream router to completely wake up. Thus, we can efficiently reduce the packet latency increase. Furthermore, as all VCs in the same input port of a router share the same DB, we can keep a minimal number of DBs powered on (on duty) and power off all of the VCs. In this way, we use minimal static power consumption to keep a certain transmission ability, which is helpful to reduce the static power consumption. The specific novel contributions of this chapter are the following:

- We propose a novel Duty Buffer structure, which can be used to replace any VC in a router. Taking the advantage of our novel DB, we propose a novel

DB-based power gating approach on VCs in a router. This approach efficiently reduces the packet latency increase caused by the power gating. By keeping a minimal number of DBs powered on, our DB-based approach also achieves a significant reduction of the static power consumption in a NoC.

- By experiments, we show that our DB-based power gating approach can effectively reduce the packet latency increase caused by the power gating. Taking a conventional router without power gating as the baseline, our DB-based power gating approach, with one flit depth of DB, increases the average packet latency only by 9.67%, which is much less than the 57% latency increase in [MKWA08] and 21.75% in [ZOG$^+$15]. Compared with the based in terms of the power consumption, our DB-based approach can save on average 52.19% of the total power consumption, which is comparable with the 59.39% saving in [MKWA08] and 57.05% in [ZOG$^+$15].

3.3 Related work

In this section, we discuss the related works on reducing the packet latency increase caused by power gating in a NoC.

As VCs are the main source of the static power consumption in a router, Zhan [ZOG$^+$15] applies the power gating on the VCs and uses the drowsy SRAM [FKM$^+$02] to build the VCs because the wakeup delay of the drowsy SRAM is only two clock cycles, thus the wakeup process is much faster. However, this approach cannot completely remove the wakeup delay and packets accumulate large wakeup delay along the whole routing path. In contrast, our approach keeps a certain transmission ability of routers when the VCs are powered off. The packets are not blocked by the powered-off VCs in the routers. Thus, even though a packet experiences multiple wakeup processes along the routing path, the packet will not accumulate large wakeup delay. Therefore, our approach can reduce more efficiently the packet latency increase.

Based on the principle of the look-ahead routing, Matsutani [MKWA08] proposes a runtime power gating approach. By sending a wakeup signal into the next router ahead of the packet injection, this approach hides a few clock cycles in the wakeup process and wakes up the powered-off router earlier. However, the number of hidden clock cycles is determined by the number of the router pipeline stages, and is insufficient to cover the entire wakeup delay. By sending the wakeup signal to the rest of the routers along the routing path, Chen [CZPP15] improves the approach in [MKWA08]. In this way, the powered-off routers can be waked up much earlier and this approach achieves almost non-blocking power gating in deterministic routing. However, this approach highly depends on the correct prediction of the routing path. If the routing path of a packet is adaptively changed, this approach may not find the correct routers to power on. Compared with [MKWA08, CZPP15], the power gating mechanism in our approach does not need the handshaking control signals (the WU signal and the PG signal in Figure 2.3) between routes to control the packet transmission. By keeping a certain transmission ability of routers when the VCs are powered off, routers in our DB-based approach still can transfer packets during the wakeup process, which can be efficiently used to significantly reduce the overall packet latency increase in both deterministic routing and adaptive routing.

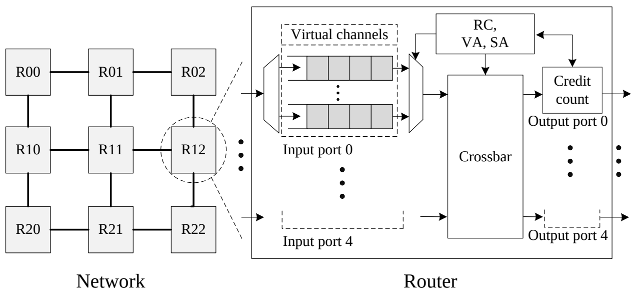
Kim proposed in [KKY11] the Flexibuffer scheme to reduce the static power consumption of the VC buffers. Based on the buffer occupancy rate, the router predictively powers on the sleeping buffers in the VCs in advance. In this way, the negative impact of the power gating on the packet latency increase is reduced. However, the scheme in [KKY11] can reduce the static power consumption, only when the VC size is large enough. This is because each VC in this approach has to keep at least $b_{min} = max(N_{wakeup}, N_{crt})$ (N_{wakeup} and N_{crt} are the number of clock cycles of the wakeup delay and credit round-trip delay, respectively) buffers powered on at run-time, which makes the flexibuffer scheme not very efficient in reducing the static power consumption. Furthermore, considering the high wakeup delay, 6-12 clock cycles [CZPP15], for NoCs with small size of VCs, such as the NoCs in [DCS$^+$14, ZOG$^+$15, CZPP15, YL16, DNSD13], the scheme in [KKY11] cannot save any power consumption. Compared with the scheme in [KKY11], the DB in our approach is shared with all VCs in the same input port, and the minimal DB size is one flit. In this way, our approach can be widely and efficiently used to reduce the static power consumption in NoCs with small and large size of VCs.

3.4 DB-based Approach

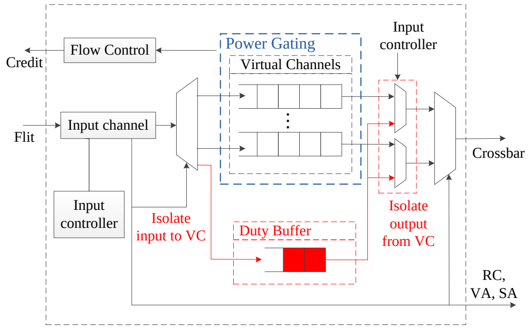
The key idea of our DB-based power gating approach is, by always having a small number of buffers powered on (on duty) in VCs, to keep a certain transmission ability of a router in the wakeup process when power gating is used. In this way, we can reduce the packet latency increase caused by the power gating. In order to achieve this goal, we have to overcome the following challenges.

- **Which VCs should be on duty?** As we have introduced in Section 2.1.4, the input VC address of a packet, which indicates where the packet will be stored, is determined in an upstream router. Therefore, a downstream router does not know when packets will arrive or which VCs will be occupied. As a consequence, it is unknown which VC should be on duty in a downstream router.

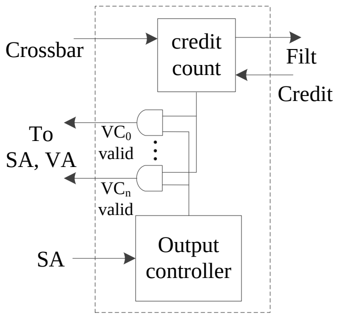
- **How to use as few buffers on duty as possible?** Keeping fewer buffers on duty in a VC is helpful to reduce the static power consumption in a NoC. However,



(a) Architecture of a NoC



(b) Extended input port



(c) Extended output port

Figure 3.1: The NoC structure and extended input port and output port.

the role of the VCs is not the same. Different classes of VCs are used to receive different packets. For example, the MESI [PP84] coherence protocol needs at least two different data VCs and one control VC to prevent a deadlock. If we want to keep some buffers in VCs on duty to reduce the packet latency increase, each VC class should have buffers on duty and ready to receive packets. As a consequence, we cannot efficiently reduce the static power consumption.

3.4.1 Input Port with Duty Buffer

In order to overcome the challenges, mentioned above, we propose the novel Duty Buffer structure shown in Figure 3.1(b) to extend the input ports of a virtual-channeled wormhole router. Compared with the conventional router in Figure 3.1(a), which is a simplified version of Figure 2.5, the following components are added to an input port: Input controller, Duty Buffer, and multiplexers. We apply power gating on the VCs as well. The Duty Buffer (DB) is a small buffer queue. The minimal size of the DB is one flit. It is always powered on and ready to receive packets at run-time.

For a downstream router, when a packet reaches an input port, but input VCs are powered-off or waking up, the input controller controls the input channel and demultiplexer to store the packet into the DB. The DB replaces the corresponding powered-off VC, which the packets should go into.

When the router tries to read the packets from the powered-off VC, the input controller controls the corresponding multiplexer to select reading the packets from the DB. By replacing the powered-off VC with DB, the router can transfer the packet as if this VC was powered on.

For an upstream router, as its corresponding downstream router can use the DB to replace the powered-off VCs to store packets, there is no need for the upstream router to block the packet and to wait for the VCs in the downstream router to completely wakeup. Thus, the upstream router still can send a packet to the downstream router, when the VCs in the downstream router are powered-off or waking up. As a result, the packet latency increase caused by the power gating on VCs is reduced.

Since the DB can replace any powered-off VC, we do not need to determine which VC should be on duty. Furthermore, as all VCs in an input port share the same DB, we do not need to keep powered-on buffers for each class of VCs. Thus, we can keep as few buffers on duty in the DB as possible, and power off as many buffers in VCs as possible. In this way, we keep a certain transmission ability with minimal static power consumption.

3.4.2 Power Gating on VCs

In our extended input port, as shown in Figure 3.1(b), the input controller uses one switch to control the power of all VCs in the port. This is because powering on all VCs at the same time is beneficial to guarantee the NoC performance. That is, the traffic load in many-core systems is bursty [DMMD09], so routers tend to use a larger number of VCs in an input port at the same time. On the other hand, the average traffic load is very low [DMMD09]. This makes a higher percentage of input VCs idle during the applications execution. So, simultaneously powering off all VCs also can reduce significantly the static power consumption.

The rest of the components in a router, for instance, the routing computation unit, the virtual channel allocation unit, the switch allocation unit, the crossbar and the output ports, are always powered on. In this way, if VCs in some input ports are powered off, the other input ports still can normally work. Packets will not be blocked in the router pipeline. Furthermore, compared with VCs, the power consumption in the rest of the router's components is much lower [CZZ+15]. Always keeping them powered on, routers will not waste much static power.

3.4.3 Power Gating Scheme

In order to correctly use our DB to reduce the latency increase during the waking up process, we have to achieve the following goals:

- **Keeping the flits order in a packet.** The flits of a packet may be separately stored in the DB and a VC when the state of VCs switches from waking up state to charging complete state. In order to keep the packet transmission correct, the transmission order of the flits in the same packet must not be changed.
- **No deadlock occurs.** In general, different class packets are allocated with different VCs. This is because a network interface has to finish processing the reply packet first, then to deal with the new request packet. If a request packet and a replay packet are stored in the same buffer queue (FIFO), the request packet may be stored in front of the reply packet and it stalls the reply packet to access the network interface. As a consequence, the request packet and the replay packet block each other and the deadlock occurs [DT04]. Therefore, in the wakeup process, we have to guarantee that only packets with the same VC address enter the input port to prevent deadlock occurrence. This VC address should be determined by the head flit of the packet, which wakes up the powered-off VCs in the downstream router.

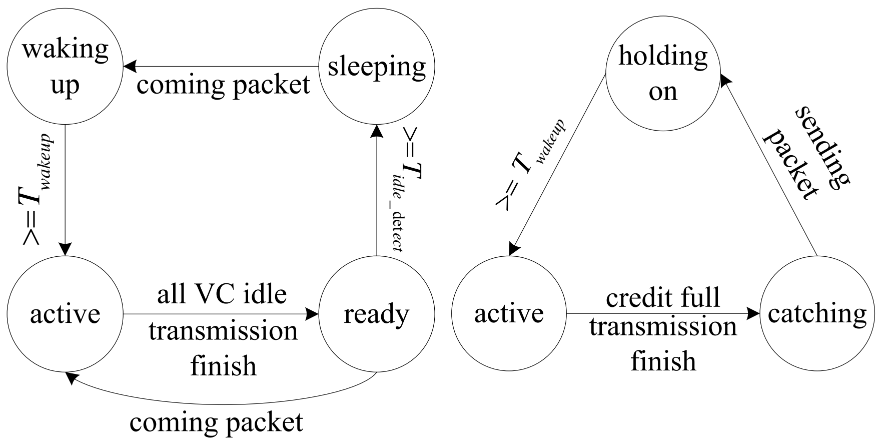
In order to achieve the aforementioned goals, the input controller and the output controller are added to a router, as shown in Figure 3.1(b) and Figure 3.1(c). In the following subsections, we introduce the working mechanisms of the input controller and the output controller.

Input Controller

In the extended input port, as shown in Figure 3.1(b), the input controller monitors the VC control to determine if it can switch off the power of VCs. The states of the input controller are shown in Figure 3.2(a).

In the **active** state, VCs in the input port are powered on. The input controller controls the input channel to inject packets into the corresponding VCs and keeps the DB idle. When all VCs and the DB are empty and the packet transmission is completed (the tail flit of the packet has left), the input port controller moves to the **ready** state and waits for T_{idle_detect} clock cycles to switch off the power of the VCs. We use the equation, $T_{idle\_detect} = T_{credit\_delay} + T_{flit\_delay}$, to compute the T_{idle_detect}, where T_{credit_delay} and T_{flit_delay} are the credit and flit transmission delaies between neighbor routers, as shown in Figure 2.4. and they are dependent on design parameters.

In the **ready** state, the VCs are still powered on and can be used to store packets. If there are incoming packets in the T_{idle_detect} period, the packets will be stored to



(a) Input controller states (b) Output controller states

Figure 3.2: Controllers for input and output ports.

the corresponding VC and the input controller returns back to the **active** state. The **ready** state is used to avoid unnecessary power gating activities when the idle time of VCs is short.

In the **sleeping** state, the power of all VCs is switched off and the VCs cannot be used to store packets. Once the head flit of a packet comes into the input port, it will be stored in the DB and the power of VCs will be switched on to charge the circuit. The input port goes into the **waking up** state.

In the **waking up** state, VCs are not stable and cannot accept packets. The incoming packets still are stored in the DB. After T_{wakeup} clock cycles, the VC charge is completed, and the state changes to the **active** state and the VCs can be used to store packets. The input controller stores the packets to the corresponding VC and stops injecting packets into the DB.

At the beginning of the **active** state, if there are any flits left in the DB, the input controller keeps replacing the output of the corresponding VC with the output of the DB until the DB is empty. In this way, the order of the flits in a packet will not be disturbed and the correct transmission is guaranteed.

Output Controller

As explained in the beginning of Section 3.4.3, to guarantee deadlock-free packet transmission during the wakeup process, an output port in a router has to send only packets with the same VC address to the downstream router. In order to achieve this,

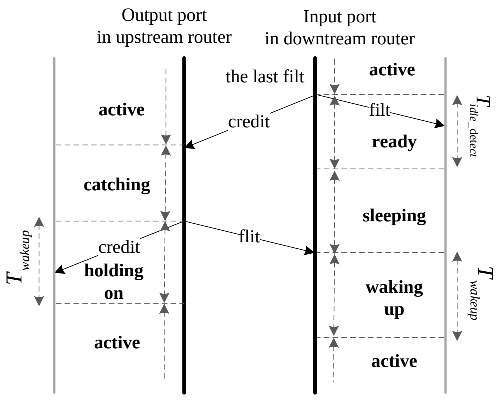


Figure 3.3: The interactive process between an output port in the upstream router and the input port in the corresponding downstream router.

the output controller and some AND gates are added to the output port, as shown in Figure 3.1(c). In this extended output port, the output controller monitors the credit counter and the grant to use the crossbar from the switch allocator unit, to identify the state of the input port in the downstream routers and controls the validation signals for the virtual channel allocator unit and the switch allocator unit, indicating which input VCs in the downstream router are available. The states of the output controller are shown in Figure 3.2(b).

In the **active** state, the credit counter is not full, or the packet transmission is not completed (the tail flit has not left). This indicates that the VCs in the corresponding input port in the downstream router are powered on. So, the upstream router can normally send packets and receive credits.

When the credit counter is full and packet transmissions are completed, the output controller moves to the **catching** state. In the **catching** state, the output controller considers that the VCs of the corresponding input port in the downstream router are powered-off, even if, at this time, this input port may be in the **ready** state. The output port allows the virtual channel allocator unit to normally allocate VCs and the switch allocator unit to grant the use of the crossbar in the **catching** state.

Once a head flit of a packet is granted by the switch allocator unit to use the crossbar, the output controller marks the allocated VC address and moves to **holding on** state. In the following T_{wakeup} clock cycles, only packets allocated to this marked VC address are allowed to use the crossbar and transferred to the downstream router. In the **holding on** state, the output controller assumes that all the flits sent are stored in the DB of the input port in the corresponding downstream router. The output con-

Table 3.1: Parameters.

topology	2D tours
router pipeline	4-stage
network size	4×4
VC count	4 VCs per port
buffer depth	4 flits per VC
packet size	1 or 8 flits, 8B/flit
T_{credit_delay}	1 clock cycle
T_{flit_delay}	1 clock cycle
BET	10 clock cycles
T_{wakeup}	10 clock cycles

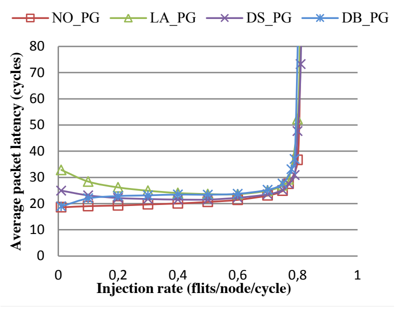
troller guarantees that the used credits does not exceed the depth of the DB. In this way, the output controller can guarantee that there is no buffer overflow in the wakeup process. After T_{wakeup} clock cycles, the output controller moves to the **active** state. The virtual channel allocator unit and the switch allocator unit can normally allocate router resource for the packets.

In better understand the interactive process between the upstream router and the downstream router, we use Figure 3.3 to the state switching in the output controller and the input port controller.

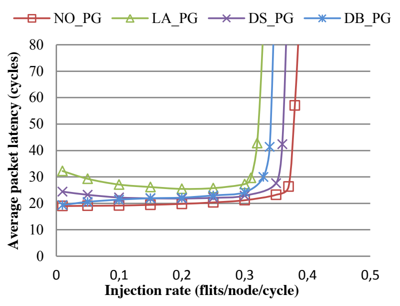
3.5 Experimental Results

In order to evaluate our approach in terms of performance and power consumption, we have implemented our approach on the cycle-accurate interconnection network simulator Booksim2.0 [JBB+13]. The parameters set in Booksim2.0 are shown in Table 3.1. Each input port of a router has four 4-flit depth VCs. The total number of buffers in a router is consistent with [DNSD13]. According to the prior works [DNSD13, ZOG+15] , the credit transmission delay T_{credit_delay} and the flit transmission delay T_{flit_delay} are set to 1 clock cycle. Based on the prior work in [DNSD13], T_{wakeup} is set to 10 clock cycles. To calculate the power consumption cost of power gating when the power of VCs is switched on and off, we set the Break even time (BET), which is the number of sleep cycles required to compensate the energy overhead for charging VCs in a power gating process. The BET is 10 clock cycles in our experiments and it is consistent with the prior work in [ZOG+15].

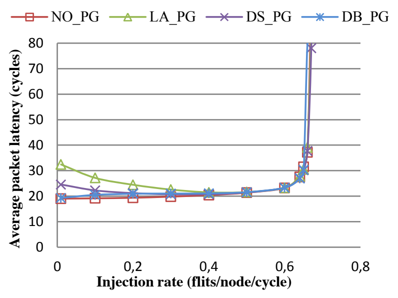
For comparison purpose, we have implemented the following schemes in Booksim2.0: (1) NO_PG is the baseline NoC. It uses the conventional four-stage pipeline router introduced in Section 2.1.4 without power gating; (2) LA_PG is a NoC using the lookahead scheme [MKWA08] to hide 4 clock cycles of the wakeup delay; (3) DS_PG is a NoC where the VCs of a router are implemented by the drowsy SRAM [ZOG+15].



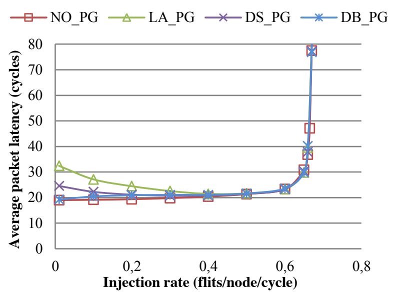
(a) Uniform



(b) Transpose



(c) Bitcomp



(d) Tornado

Figure 3.4: Average packet latency across full range of workloads.

In the active state, the drowsy SRAM has the same power consumption as the conventional SRAM. When staying in the drowsy state, it only consumes about 10% of the SRAM static power consumption and has 2 clock cycles wakeup delay; (4) DB_PG is a NoC using our DB-based approach, presented in Section 3.4, with different depths of the DB.

3.5.1 Evaluation on Synthetic Workloads

In order to explore our DB_PG behavior under a wider range of packet injection rates, in this section, we evaluate the performance of NO_PG, LA_PG, DS_PG and our DB_PG under synthetic traffic patterns. Booksim2.0 provides abundant synthetic traffic patterns. We select four synthetic traffic patterns: 1) Uniform random: packets' destinations are randomly selected; 2) Transpose: packets from source node (x, y) are sent to destination node (y, x); 3) Bitcomp: packets from (x, y) are sent to $(N - x, N - y)$, N is the number of nodes in the X and Y dimensions of a NoC; 4) Tornado: packets from (x, y) are sent to $(x + \frac{N}{2} - 1, y)$. In this experiment, 1-flit packets are injected to the NoCs. Figure 3.4 shows the average packet latency curves with different injection rates under the different traffic patterns. Since DB_PG with different depths of the DB has similar trend in terms of the average packet latency curve, in order to clearly show the experimental results, in Figure 3.4, we only show our DB_PG with DB of depth 1.

The zero-load latency is an important performance indicator for a NoC. As shown in Figure 3.4, when the injection rate is almost zero (the injection is about 0.01 packets/node/cycle), where the packet latency is close to the zero-load latency, our DB_PG has less packet latency than LA_PG and DS_PG. This is because, when the injection rate is low, most of the input ports of routers are idle. A packet experiences multiple wakeup processes along the routing path. As a consequence, a packet in LA_PG and DS_PG accumulates a large wakeup delay. In contrast, by keeping a certain transmission ability in the wakeup process, our proposed DB_PG can avoid the situation where a packet accumulates too much latency.

With the injection rate increasing from 0 packets/node/cycle to about 0.2 packets/node/cycle, as shown in Figure 3.4, our DB_PG still has less packet latency increase than LA_PG and DS_PG. However, the packet latency in our DB_PG slowly increases, while the packet latency in LA_PG and DS_PG decreases. This is because, with the injection rate increasing, more and more input ports of routers are always busy and cannot be powered off. The probability of accumulating a larger wakeup delay becomes lower. On the other hand, multiple packets may compete for a transfer to a downstream router in the same wakeup process, but our DB only can replace one VC in a signal wakeup process in order to avoid deadlock occurrence. As a consequence, in the wakeup process, some of the packets are blocked in our DB_PG.

When the injection rate further increases and is close to the saturation injection rate where the packet latency sharply increases, as shown in Figure 3.4(a) and Figure 3.4(b), the packet latency in the LA_PG and DB_PG is higher than that in DS_PG, because, in this situation, the wakeup delay in a single wakeup process becomes the main reason for the packet latency increase. The LA_PG and DB_PG have higher wakeup delay than DS_PG. However, because of the DB structure, our DB_PG outperforms LA_PG.

Based on the results of the synthetic traffic patterns, when the injection is below 0.2 packets/node/cycle, our DB_PG efficiently prevents a packet to accumulate a large wakeup delay along the routing path and achieves better performance than LA_PG and DS_PG. However, when the injection rate is close to the saturation injection rate, in some traffic patterns, our DB_PG causes a little bit performance loss in terms of the average packet latency. Considering that the injection rate of real applications [DMMD09] is much lower than 0.2 packets/node/cycle, our DB_PG can be widely used to reduce the packet latency increase caused by the power gating.

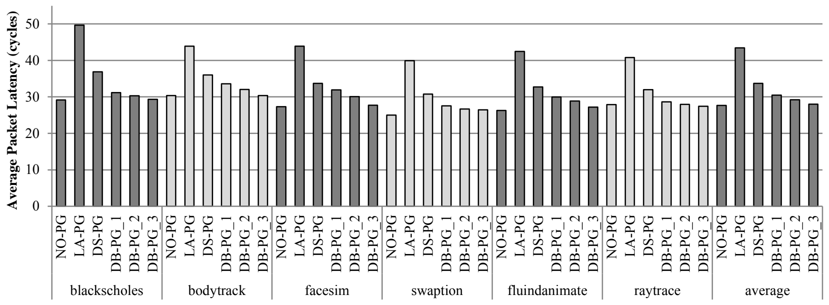


Figure 3.5: The average packet latency.

3.5.2 Evaluation on Real Application Workloads

In order to evaluate our DB-based approach and compare it with the other approaches mentioned above, on real workloads of applications, we use six applications from the Parsec [BKSL08] benchmark suite: blockscholes, bodytrack, facesim, swaption, fluindanimate, and raytrace. We use Synfull [BJ14] to capture the traffic behaviour of these applications. Synfull generates packets and feeds packets to Booksim 2.0 to evaluate the network performance. Based on the collected data from Booksim2.0, we use Dsent [SCK$^+$12] to compute the power consumption under the 45nm technology, the voltage of $1V$, and the NoC frequency of $1GHz$.

Effect on Performance

Figure 3.5 shows the average packet latency for the six different real application workloads. The seventh set of bars in Figure 3.5 gives the average result over these six applications.

We use the average packet latency to measure the network performance. The LA_PG hides 4 cycles of the wakeup delay by sending the wakeup signal in advance. However, compared with the baseline NO_PG, it still incurs an average of 57% (about 16 cycles) packet latency increase throughout these six applications. The latency in the DS_PG also increases by 21.75% (about 6 cycles) on average, even though the wakeup delay in the DS_PG is only 2 clock cycles. These results indicate that packets experience over three wakeup processes on average and the cumulative wakeup delay is the main reason for the performance degradation.

In contrast, the average packet latency in our DB_PG only increases by 9.67%, 5.67%, and 2.02% with 1, 2, and 3 flits depth of the Duty Buffer, respectively. Compared with the LA_PG and the DS_PG, our DB_PG approach, with only one flit depth of the DB, achieves 47.33% and 12.08% less packet latency increase, respectively.

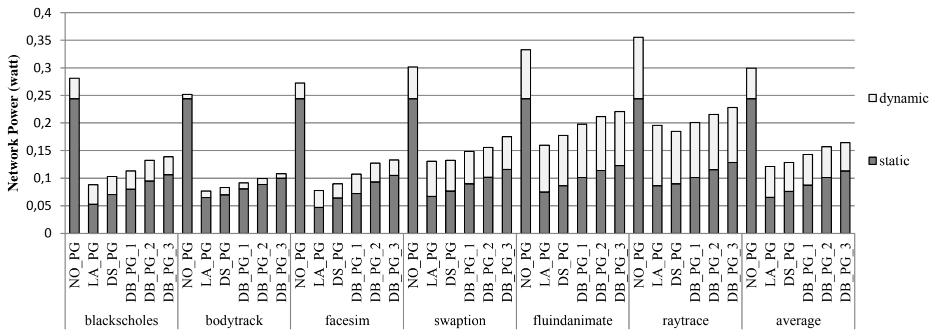


Figure 3.6: The breakdown of the NoC power consumption.

This is because our DB can replace any sleeping VC in the wakeup process to store packets. The upstream routers do not need to block packets and wait for the VCs to completely waked up. Furthermore, powering on all of the VCs in an input port at the same times can effectively deal with the bursty traffic load of real applications.

The depth of our DB also has an obvious effect on the network performance, as shown in Figure 3.5. Since we use a five-stage pipelined router, there is a notable pipeline delay and credit round-trip delay in the router. In our DB_PG with the smallest DB (DB_PG_1 in Figure 3.5), the packet transmission may be intermittently interrupted because of the credit round-trip delay between routers. In addition, as the DB only can replace one VC in a signal wakeup process, packets with different VC address may be blocked in the upstream routers, which also causes packet latency increase.

Effect on Power Consumption

Figure 3.6 shows the breakdown of the NoC power consumption across the six benchmarks and the seventh set of bars shows the average over these six benchmarks. The network power is broken down into dynamic and static power consumption.

Compared with the baseline NO_PG, the LA_PG and DS_PG can reduce an average of 73.14%, and 68.83% of the static power consumption, respectively. The static power consumption in our DB_PG with 1, 2, and 3 flits depth of the DBs is reduced by 64.11%, 58.49% and 53.63% on average, respectively. For the total power consumption, compared with the baseline NO_PG, the LA_PG and DS_PG reduce the total NoC power consumption by 59.39% and 57.05%, respectively. Our DB_PG with different DB depths reduces with 52.19%, 47.55%, and 45.14% the total power consumption, which is comparable to the LA_PG and DS_PG.

It is obvious that we can simply realize a non-blocking power gating scheme by combining the LA_PG with the DS_PG. However, in the sleeping state, the static

Table 3.2: Area of router components.

	NO_PG	DB_PG_1	DB_PG_2	DB_PG_3
input port ($\times 5$)	0.0919	0.0973	0.0992	0.1025
crossbar+VA+SA	0.0061	0.0061	0.0061	0.0061
output port ($\times 5$)	0.0065	0.0072	0.0072	0.0072
total (mm^2)	0.1045	0.1106	0.1125	0.1158

power consumption of the drowse SRAM increases with the increase of the total number of buffers in a router. As a consequence, when a NoC has a large number of buffers, the drowse SRAMs still cause significant static power consumption in the sleeping state. While, in our approach, the static power consumption in the sleeping state is only determined by the number of DBs (instead of the total number of buffers), which presents better scalability and is more suitable for a NoC with a larger number of buffers.

Effect on Area

In order to evaluate the area overhead of our DB_PG scheme compared to NO_PG, we use Synopsys Design Compiler to synthesize the routers used in our experiments under the 45nm NanGate Open Cell Library [Sil].

The area of each component is shown in Table 3.2. Compared with the baseline router used in the NO_PG, the routers used in our DB_PG_1, DB_PG_2 and DB_PG_3 cause about 5.76%, 7.65% and 10.73% area increase, respectively. Most of the area overhead in our DB_PG is contributed by the duty buffers and multiplexers in the input ports, while the area overhead of the input controller is very low. This is because the input controller is made up of simple logic circuits and a small number of registers, that do not cause large area overhead.

As our DB_PG has no influence on the crossbar, VA and SA, there is no area overhead on these components. Compared with the output ports in the baseline router (No_PG), the output ports in our DB_PG have 9.8% area overhead. However, these area overhead takes negligible percentage of the total router area because the structure of the output port controller is simple logic circuits and only contains a small number of registers.

3.6 Discussion

In this work, we proposed a novel Duty Buffer structure and DB-based power gating approach to reduce the packet latency increase caused by power gating in a NoC. As the DB can replace any VC in the input port, the input port can power off all VCs without the need to consider which VC will be used in the future. In this way, we can keep minimal number of buffers "on duty" to reduce the packet latency increase caused by the power gating, and power off most of the VCs to reduce the static power consumption. The experimental results show that our approach outperforms the lookahead and drowsy SRAM approaches. With a small amount of additional hardware overhead, our DB-based approach can efficiently reduce the static power consumption, which is comparable with the lookahead and drowsy SRAM approaches. In later experimental results, presented in Chapter 4 and Chapter 5, we show that our DB-based power gating approach even can be more effective on reducing the power consumption under a NoC configuration with more VCs. However, being a fine-grained power gating approach, our DB-based power gating needs to separately switch the power of each input port in a router. Thus, some times packets may experience more power gating processes, which may result in an extra packet latency increase.