

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

A variable frequency link for a power-aware network-on-chip (NoC)

Seung Eun Lee*, Nader Bagherzadeh

Department of EECS, University of California-Irvine, Irvine, CA 92697, USA

ARTICLE INFO

Article history:

Received 2 February 2008
 Received in revised form
 22 January 2009
 Accepted 25 January 2009

PACS:

85.40.e

Keywords:

Interconnection network
 Network-on-chip (NoC)
 Dynamic frequency scaling
 Power optimization
 System-on-chip (SoC)

ABSTRACT

Although the technology scaling has enabled designers to integrate a large number of processors onto a single chip realizing chip multi-processor (CMP), problems arising from technology scaling have made power reduction an important design issue. Since interconnection networks dissipate a significant portion of the total system power budget, it is desirable to consider interconnection network's power efficiency when designing CMP. In this paper, we present a variable frequency link for a power-aware interconnection network using the clock boosting mechanism, and apply a dynamic frequency scaling (DFS) policy, that judiciously adjusts link frequency based on link utilization parameter. Experimental result shows that history-based DFS successfully adjusts link frequency to track actual link utilization over time, demonstrating the feasibility of the proposed link as a power-aware interconnection network for system-on-chip (SoC).

© 2009 Elsevier B.V. All rights reserved.

1. Introduction

The technology scaling has enabled designers to integrate a large number of processors onto a single chip, realizing chip multi-processor (CMP). High performance CMP architectures have been gaining the attention of high performance computing community in the past few years. As the demand for network bandwidth increases for CMP, the idea of network-on-chip (NoC) becomes more promising because of performance, power, and scalability requirements for an SoC design [1].

Although today's processors are much faster and far more versatile than their predecessors using high-speed circuits and parallel processing, they also consume a lot of power. Moreover, an interconnection network dissipates a significant fraction of the total system power budget. For instance, the MIT raw on-chip network consumes 36% of the total chip power and Alpha 21364 microprocessor dissipates 20% of power in interconnection network [2]. Therefore, an interconnection network must be designed to be power-aware.

In this paper, we motivate the use of dynamic frequency scaling (DFS) link, where the frequency is dynamically adjusted to minimize power dissipation while maintaining the performance demands. First, we propose a novel DFS link which adopts a clock boosting mechanism [3], providing fast response time for frequency transitions and low hardware overhead. Next, a DFS policy is introduced that includes the link utilization estimator,

DFS algorithm, and link controller. Finally, the DFS policy is applied to the proposed DFS link, demonstrating the power saving in an on-chip interconnection network. To the best of our knowledge, this is the first investigation of power reduction for on-chip interconnection network based on the clock boosting mechanism.

The rest of this paper is organized as follows. Section 2 addresses existing work on power saving for on-chip interconnection network. The proposed DFS link based on the clock boosting mechanism is introduced in Section 3. The implementation and the experimental results are presented in Sections 4 and 5, respectively. Finally, conclusions are drawn in Section 6.

2. Backgrounds

2.1. Dynamic voltage/frequency scaling

A communication link in NoC is capable of scaling power consumption gracefully commensurate with traffic workload. This scalability allows for the efficient execution of energy-agile algorithms. Suppose that a link can be clocked at any nominal rate up to certain maximum value. This implies that different levels of power will be consumed for different clock frequencies. One option would be to clock all the links at the same rate to meet the throughput requirements. However, if there was only one link in the design that required to be clocked at a high rate, the other links could be clocked at a lower rate, consuming less power.

The total power consumption in an SoC is the combination of dynamic and static sources. In this paper, our focus is on the

* Corresponding author. Tel.: +1949 422 7130.

E-mail addresses: seunglee@uci.edu (S.E. Lee), nader@uci.edu (N. Bagherzadeh).

dynamic power consumption which arises from circuit switching activity, due to charging and discharging of the switched capacitance. The dynamic power consumption depends on four parameters: a switching activity factor (α), physical capacitance (C), supply voltage (V), and the clock frequency (f)

$$P_D = \frac{1}{2}\alpha CV^2f \quad (1)$$

$$f_{max} = \eta \frac{(V - V_{th})^\beta}{V} \quad (2)$$

Eq. (2) establishes the relationship between the supply voltage V and the maximum operating frequency f_{max} , where V_{th} is the threshold voltage, and η and β are experimentally derived constants.

Dynamic power consumption can be reduced by lowering the supply voltage. This requires reducing the clock frequency accordingly to compensate for the additional gate delay due to the lower voltage. The use of this approach in run-time, which is called dynamic voltage scaling (DVS), addresses the problem of how to adjust the supply voltage and clock frequency of the link according to the traffic level. The basic idea is that because of high variance in network traffic, when a link is under-utilized, the link can be slowed down without affecting performance. However, DVS requires thousands of clock cycles during transition between voltage levels and additional hardware overhead for each link.

The other way to manage power consumption is DFS. DFS only adapts the system clock frequency by setting all links in the network to the same voltage, but it does not always reduce the total energy consumption. For instance, the power consumed by a network can be reduced by halving the operating clock frequency, but if it takes as long to forward the same amount of data, the total energy consumed will be similar. DFS is valid when the target system does not support DVS or the goal is to reduce peak or average power dissipation, indirectly reducing the chip's temperature [4]. An alternative to save link power is to add hardware such that a link can be powered down when it is not used heavily.

2.2. Related works

System level power management has been applied to some interconnection networks. Wei and Kim proposed chip-to-chip parallel [5] and serial [6] link design techniques where links can operate at different voltage and frequency levels. When link frequency is adjusted, supply voltage can track to the lower suitable value. Although this link was not designed for both dynamic voltage and frequency settings, previously the link architecture was used for DVS link model.

There are three kinds of approaches for DVS. One is an on-line scheme which adjusts the link speed dynamically, based on a hardware prediction mechanism by observing past link traffic activities. Shang et al. [7] developed a history-based DVS policy which adjusts operating voltage and clock frequency of a link according to the utilization of link and input buffer. Worm et al. [8] proposed an adaptive low-power transmission scheme for on-chip networks. They minimized the energy required for reliable communication, while satisfying QoS constraints. One of the potential problems with hardware prediction scheme is that a misprediction of traffic can be costly from performance and power perspectives.

Li et al. [9] proposed a compiler-driven approach where a compiler analyzes application code and extracts communication patterns among parallel processors. These patterns and the inherent data dependency information of the underlying code help the compiler decide the optimal voltage/frequency to be used

for communication links at a given time frame. Shin and Kim [10] proposed an off-line link speed assignment algorithm for energy-efficient NoC. Given the task graph of a periodic real-time application, the algorithm assigns an appropriate communication speed to each link, while guaranteeing the timing constraints of real-time applications.

Soteriou et al. [11] proposed a software-directed methodology that extends parallel compiler flow in order to construct a power-aware interconnection network, by combining both on-line and off-line approaches. However, current DVS techniques require not only thousands of clock cycles to shift between voltage levels, limiting their ability to respond to high frequency changes in network bandwidth demands [12], but also additional hardware overhead such as transmitter, receiver, PLL, and adaptive power supply regulator for each link.

Kim et al. [13] proposed dynamic link shutdown (DLS), which powers down links intelligently when their utilizations are below a certain threshold level and a subset of highly used links can provide connectivity in the network. An adaptive routing strategy that intelligently uses a subset of links for communication was proposed, thereby facilitating DLS for minimizing energy consumption. Soteriou and Peh [2] explored the design space for communication channel turn-on/off based on a dynamic power management technique depending on hardware counter measurement obtained from the network during run-time. Chen et al. [14] introduced a compiler-directed approach, which increases the idle periods of communication channels by reusing the same set of channels for as many communication messages as possible. Li et al. [15] proposed a compiler-directed technique in order to turn off the communication channels to reduce NoC energy consumption. Even though it saves power significantly during idle period, it has reactivation penalty including delay and additional power consumption during a transition.

Hsu [16] saved 30% of power consumption in the MPEG core by applying DFS power management mechanism using only three frequency levels (25, 50, and 100 MHz). However, DFS was applied to a core, not to interconnection network, in a tile-based NoC architecture. To the best of our knowledge, this paper is the first proposal which addresses DFS for interconnection network.

The novel contributions of our work are:

- (1) a DFS link proposal for on-chip interconnection network which offers not only fast response time reducing the frequency transition penalty, but also reduces hardware cost, as compared to DVS link, suitable for system integration;
- (2) use of narrow control period, as compared to conventional DVS control, reducing misprediction penalty that occurs in a hardware prediction scheme by adjusting the frequency more often;
- (3) implementation of a variable frequency link that judiciously adjusts link frequency based on the link utilization estimation, reducing power consumption.

3. Variable frequency link

The clock boosting router was proposed to increase throughput and reduce latency of an adaptive wormhole router [3]. The key idea of clock boosting mechanism is *the use of different clocks in a head flit and body flits* because body flits can continue advancing along the reserved path that is already established by the head flit, while the head flit requires the support of complex logic, increasing critical path. Thus, it reduces latency and increases throughput of a router by applying faster clock frequency to a boosting clock in order to forward body flits.

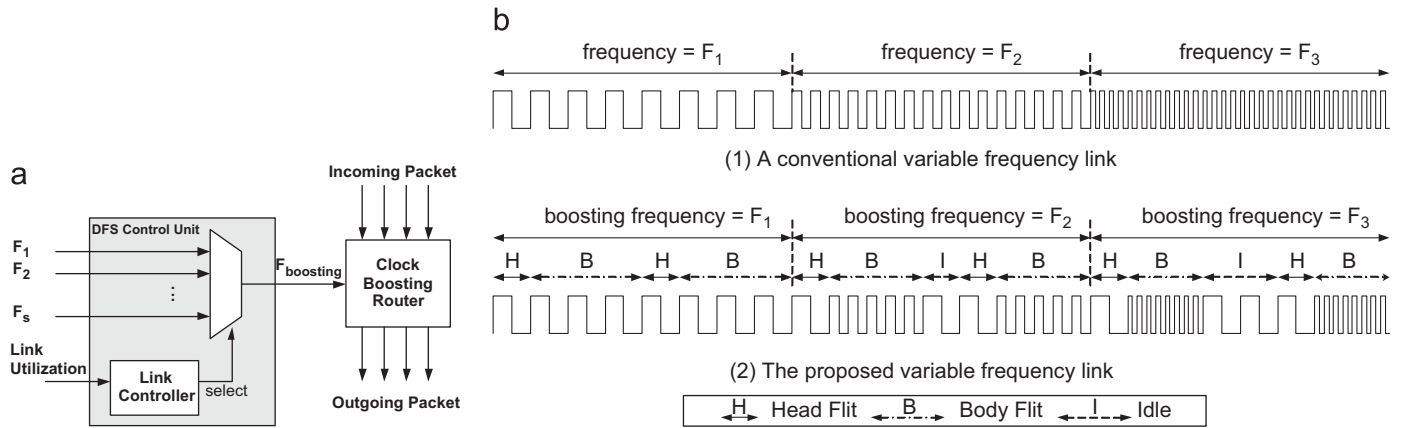


Fig. 1. (a) Architecture of a DFS link, (b) time-space diagram showing the clock domain transition in DFS link: (1) a conventional variable frequency link and (2) the proposed variable frequency link.

DFS only adapts the system clock frequency by setting all links in the network to the same voltage. The clock boosting router can be modified to support a variable frequency link that is applicable for DFS with negligible hardware cost and fast response time to frequency changes. In addition, the operating frequency of a system is not limited by the critical path of the route decision logic because it only changes clock frequency for the body flit transmission. Thus, the proposed method not only provides variable frequency link but also increases interconnection network performance. Also, fast response time of the clock domain variations makes it possible to use narrow control period for DFS, where clock frequency is adjusted more frequently. Fig. 1(a) shows an example of the proposed variable frequency link using clock boosting mechanism. The system has multiple clock frequencies represented by F_i . Link controller selects boosting clock frequency for the clock boosting router among supported clock frequencies by using link utilization level. Fig. 1(b) shows the time-space diagram for variable frequency links. In this example, the link supports three different frequencies (F_1, F_2 , and F_3). A conventional variable frequency link changes clock domain for the entire control period, while the proposed variable frequency link applies different clock frequencies only to the body flit transmission. The original clock frequency (F_1 in this example) is still used for the head flit transmission as well as for idle cycles.

In this paper, we use multiple clock frequencies ($1\times, 2\times$ and $4\times$) as the boosting clock frequency for the body flit transmission, in order to reduce implementation complexity. Table 1 summarizes the characteristics of a single link with different boosting clock frequencies using TSMC 90 nm technology. By boosting clock frequency, throughput of a link is increased at the expense of more power consumption, demonstrating the application of a DFS link for an NoC. The power saving is from the adaptation of boosting clock frequency according to the traffic load at the given time period. Because head flit is a small part of a packet, we can achieve power saving by adjusting the operating frequency for body flit transmission. Even though the DFS link in this paper supports three different levels of clock frequency, experimental results show power saving potential for any interconnection network.

4. History-based DFS

4.1. Problem formulation

The DFS link allows different frequencies: ϕ_1, ϕ_2, \dots , and ϕ_s . In addition, changing frequencies does not incur too much overhead.

Table 1

Characteristics of clock boosting router.

Boost clock (MHz)	Throughput (Gbps)	Dynamic power (mW)	Total power (mW)
100 ($1\times$)	5.8	1.69	1.84
200 ($2\times$)	9.6	2.51	2.66
400 ($4\times$)	12.8	3.54	3.69

Table 2

Symbols and meanings of the parameters used for this study.

T_c	Control period for the DFS link control
ϕ_1	Operating frequency for header flit transmission and idle cycle
ϕ_q	One of the available frequencies, $\phi_1 < \phi_2 < \dots < \phi_s$
w_{H_i}	Number of head flits in the i th workload
w_{B_i}	Number of body flits in the i th workload
w_{I_i}	Number of idle cycles in the i th workload
δ_i	Packets forwarding time for the i th workload
f_i	Boosting frequency in the i th period

There are two repetitive jobs: making a path decision (j_h) for a head flit and forwarding body flits (j_b). A path decision is processed before forwarding body flits ($j_h \rightarrow j_b$) (Table 2 summarizes parameters used for this study).

DFS link control is performed for each control period T_c . The i th period is the time interval of $[(i-1)T_c, iT_c]$. Let $N = \{1, 2, \dots, n\}$. We use f_i to indicate the frequency during the i th period, $f_i \in \{\phi_1, \phi_2, \dots, \phi_s\}$ and $i \in N$. Suppose w_{H_i} and w_{B_i} are the number of head and body flits arriving at a link during i th period, respectively. It takes w_{H_i}/ϕ_1 to make a path decision at frequency ϕ_1 and w_{B_i}/f_i to forward body flits at frequency f_i . Thus, packets forwarding of the i th workload takes δ_i time as follows:

$$\delta_i = \frac{w_{H_i}}{\phi_1} + \frac{w_{B_i}}{f_i} + \frac{w_{I_i}}{\phi_1} \quad (3)$$

From Eq. (1), dynamic power consumption in i th period is

$$P_{D_i} = \frac{1}{2} \alpha C V^2 \phi_1 \frac{(w_{H_i} + w_{H_i} + w_{B_i})f_i}{(w_{H_i} + w_{H_i})f_i + w_{B_i}\phi_1} \quad (4)$$

Therefore, the problem of power minimization in DFS is to find a frequency (the value of f_i for $i \in N$) to minimize power consumption, while satisfying δ_i constraint.

4.2. Link utilization estimation

In applying DFS to a system, how to predict future workload with reasonable accuracy is a critical problem. This requires knowing how many packets will traverse a link at any given time. Two issues complicate this problem. First, it is not always possible to accurately predict future traffic activities. Secondly, a sub-system can be preempted at arbitrary times due to user and I/O device requests, varying traffic beyond what was originally predicted. In order to estimate future work load, we adopt link utilization as an indicator, which is a direct measure of traffic through a link in each unit time. Lower link utilization reflects the more idle cycles in a link caused by network congestion with heavy traffic or sparse workload in the incoming port. Conversely, higher link utilization implies that more active cycles in a link passing flits to the destination router.

The link utilization is measured by sampling a link at a given time during a pre-defined control period (T_c). The direct link utilization is defined, where k denotes the number of samples in T_c time period:

$$U_L(n) = \frac{\sum_{t=1}^k u(t)}{k}$$

$$u(t) = \begin{cases} 1 & \text{if there is link traffic in cycle } t \\ 0 & \text{if there is no link traffic in cycle } t \end{cases} \quad (5)$$

The direct estimator only measures the link utilization whether a link is occupied or not. It does not consider the number of flits traversing through a link during the given time. For instance, even though the link utilizations are the same in time durations Δt_1 and Δt_2 , the number of flits passing through the link can be different according to the boosting clock frequency of the router at those times.

Direct estimation can be realized with a counter, reducing the complexity of the estimator and additional hardware overhead caused by the DFS link controller. A counter at each output port gathers the total number of cycles that are used to pass a flit in each control period by counting $u(t)$ with $4 \times$ clock for accurate measurement (Fig. 2(a)). Function $u(t)$ is assigned to the write enable signal of the router, and the counter value is sampled in each control period to complete the measurement of the link utilization.

Network workload exhibits transient fluctuation and long-term transitions. In order to filter out transient fluctuations from link utilization and to predict future communication workload, distributed history-based DVS policy was proposed [7].

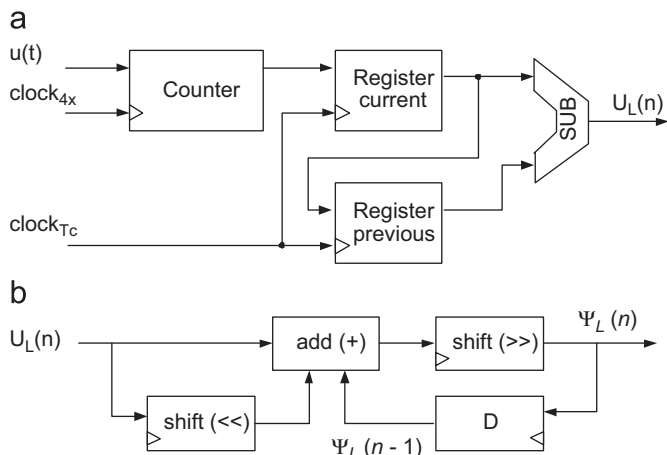


Fig. 2. History-based link utilization estimator: (a) direct link utilization estimator and (b) exponential weighted average calculator.

History-based link estimator uses exponential weighted average utilization to combine current ($U_L(n)$) and past ($\Psi_L(n - 1)$) link utilization history, smoothing and predicting future link utilization $\Psi_L(n)$ as follows:

$$\Psi_L(n) = \frac{weight \times U_L(n) + \Psi_L(n - 1)}{weight + 1} \quad (6)$$

where $\Psi_L(0) = \psi_0$, $i \in N$, and $weight$ is the contribution factor of current link utilization level to the history-based link estimator.

The hardware overhead is an important factor for the design of the estimator. Soteriou [2] realized the history-based estimator with two shifters and an adder by setting $weight$ equal to 3, reducing additional hardware overhead caused by the prediction mechanism. Fig. 2(b) shows the hardware circuit for the exponential weighted average. The result of direct estimator is fed to the exponential weighted average calculator to predict the link utilization. The history-based estimator is a cascade of direct link utilization estimator and an exponential average calculator.

4.3. DFS algorithm

Given the link utilization, the DFS algorithm dynamically adapts its frequency to achieve power savings with minimal impact on performance. It prescribes whether to increase boosting clock frequency to higher level, decrease boosting clock frequency to lower level, or do nothing. Even though the link utilization estimator predicts correctly the workload, determining how fast to run the network is nontrivial. The algorithm controlling DFS link trades off power and performance. Intuitively, if a link utilization is going to be high ($\Psi_L \geq \pi_u$), the boosting clock frequency will be increased. On the contrary, when a link utilization falls below the threshold value ($\Psi_L < \pi_l$), the boosting clock frequency is reduced. The threshold values (π_u and π_l are the threshold values to increase and decrease the boosting frequency, respectively) can be set to a single value for π_u and π_l for the simplest method. Also, multiple thresholds can be set corresponding to each state (three sets of thresholds from $\pi_{l_{1x}}$ to $\pi_{l_{4x}}$ and $\pi_{u_{1x}}$ to $\pi_{u_{4x}}$). In addition, threshold values can be pre-defined in design-time, or optimized in run-time. The pseudo-code of our DFS policy is shown in Algorithm 1.

Algorithm 1.

Dynamic frequency scaling

```

while (DFS enable) do
     $\Psi_L(n) = (W \times U_L(n) + \Psi_L(n - 1)) / (W + 1)$ 
    if  $\Psi_L(n) \geq \pi_u$  then
        Increase boosting clock frequency ( $\uparrow$ )
    else if  $\Psi_L(n) < \pi_l$  then
        Decrease boosting clock frequency ( $\downarrow$ )
    else
        Maintain current boosting clock frequency ( $-$ )
    end if
end while
    
```

4.4. Link controller

The link controller is implemented with a Moore machine. Each state represents the boosting clock frequency such as f_{1x} , f_{2x} , and f_{4x} with a two-bit value, and the machine output, equal to the state value, is passed on the clock domain multiplexer. In our link controller, there is no change between state (f_{1x}) and state (f_{4x}). Clock domain transition only occurs between adjacent clock frequencies. We assign the state values such that the Hamming distance between state transitions is 1. Clock is the most

Table 3
Physical characteristics of a clock boosting router, an 8-depth FIFO, and history-based DFS controllers by varying control period from 8 to 128 clock cycles.

	Boosting router	FIFO 8-depth	History-based DFS controller (cycles)				
			8	16	32	64	128
Voltage (V)	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Freq. (GHz)	0.425	1.72	4.00	3.33	2.94	2.56	2.38
Area (μm^2)	16,609	14,806	728	862	997	1138	1278

important and sensitive signal in a system and glitches between clock domain transitions make the system unstable, resulting in erroneous signals. To ensure constancy of the clock phase during clock domain changes, we set the control period to multiples of the clock period of the original clock frequency; and the link control function is performed in each control period.

4.5. Physical characteristics

A logic description of our router's component has been obtained by the synthesis tool from the Synopsys™ using TSMC 90 nm technology. Table 3 summarizes the physical characteristics of each element. Clock boosting router operates up to 425.5 MHz and the FIFO operates up to 1.72 GHz. For this paper, FIFO depth is set to eight. The wider control period imposes more area cost in DFS control logic because it requires a wider register, an adder, and a shifter.

The overall design including a router, eight FIFOs with size 8 and six DFS controllers occupies an area of approximately 0.143 mm² in 90 nm technology.¹ However, the ARM11 MPCore™ and PowerPC™ E405, which provide multi-CPU designs, occupy 1.8 and 2.0 mm² in the same technology, respectively [17,18]. If the link was integrated within a CMP as an interconnection network, the area overhead imposed by the network would be reasonable showing the feasibility.

5. Experimental results

5.1. Experimental setup

In order to evaluate the performance of the proposed DFS link, each component was modelled in Verilog™HDL. Fig. 3 shows the experimental setup for evaluating characteristics of a single DFS link. For this experiment, the source router sends packets to the sink router, and a FIFO is located between the routers. There are four arriving packets to the source router from North, West, South, and Internal node that should be forwarded to the East output port. The power consumed by an intermediate FIFO depends largely on the amount of buffering and the architecture. The depth of a FIFO between links is fixed to eight in this experiment. For the measurement of throughput and adjusting incoming traffic, we adopted a standard interconnection network measurement setup [19], where the packet generation is placed in front of an infinite depth source queue and an input timing of each packet is measured whenever it is generated.

The power consumption of the interconnection network was extracted using 90 nm technology. The RTL description is synthesized to the gate level net-list with Synopsys Design Compiler™ [20] using technology library. As part of this step, physical information

¹ The clock boosting router has two disjoint sub-networks for the west-to-east and east-to-west for deadlock free operation, utilizing eight input FIFOs [3].

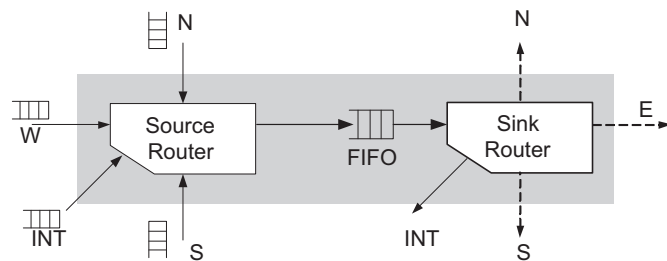


Fig. 3. Experimental setup.

such as RC parasitic value files (SPEF), standard delay format (SDF) and design constraints file (SDC) are also generated to be used for gate level power analysis. The gate level simulation extracts latency information of each packet and generates a value change dump (VCD) file for the power analysis. Power analysis with Synopsys PrimeTime™ PX tool [20] creates nanosecond detailed power waveform using switching and physical information.

Fig. 4(a) shows the histogram of the workload of a link over 24 μs obtained by using benchmarks from the SPLASH-2 [21] suite for a 7 \times 7 network. It confirms the presence of high variance in network traffic. In this analysis, power consumption under a given traffic pattern is investigated. Even though this traffic pattern cannot realistically reflect all types of traffic that will traverse the network, using this traffic pattern provides a reasonable measurement for the performance of this method.

5.2. DFS link characteristics

The DFS link characteristics for each boosting clock frequency are obtained by simulation under the given workload (see Table 4). The 1 \times boosting router finishes the entire packet transmission in 24.34 μs , spending more time than 2 \times and 4 \times boosting router. It also has the highest average and peak latency. The 2 \times boosting router reduces average latency by about 81% at the expense of 16% more dynamic power in contrast to the 1 \times boosting router. Similarly, 4 \times boosting router is much better as compared with the 1 \times boosting router in terms of latency; however, it consumes 21% more dynamic power, reducing average latency to 87%. It also reduces the average latency around 32% at the expense of only 5.1% more dynamic power in comparison with 2 \times boosting router.² These experimental results demonstrate the feasibility of clock boosting router for the DFS link for a power-aware on-chip interconnection network for an NoC platform.

5.3. History-based DFS

The history-based DFS policy was applied to the DFS link. Threshold values for the link controllers, π_u and π_l , were set to 80% and 50%, respectively.

Fig. 4 shows the simulation result of the history-based DFS policy when control period (T_c) is eight cycles. Link utilization estimator predicts future workload based on the history of workload (Fig. 4(b)). The DFS policy dynamically adjusts link frequency according to the link utilization level (Fig. 4(c)). Fig. 4(d) shows that history-based DFS successfully adjusts link frequency to track actual link utilization over time, reducing

² From Eq. (4), dynamic power of 2 \times and 4 \times operations are 26% and 40% more than the power of 1 \times operation ($P_{D_{2x}} = 1.26 \cdot P_{D_{1x}}$ and $P_{D_{4x}} = 1.4 \cdot P_{D_{1x}}$), by assuming $P_{D_{ix}} = \frac{1}{2} \alpha C V^2 \phi_i$. The experimental results show less power values because idle operation and head flit transmission have different switching activities even though they operate with the same frequency. Increasing boosting clock frequency results in more idle cycles since the body flits are transmitted with higher frequency.

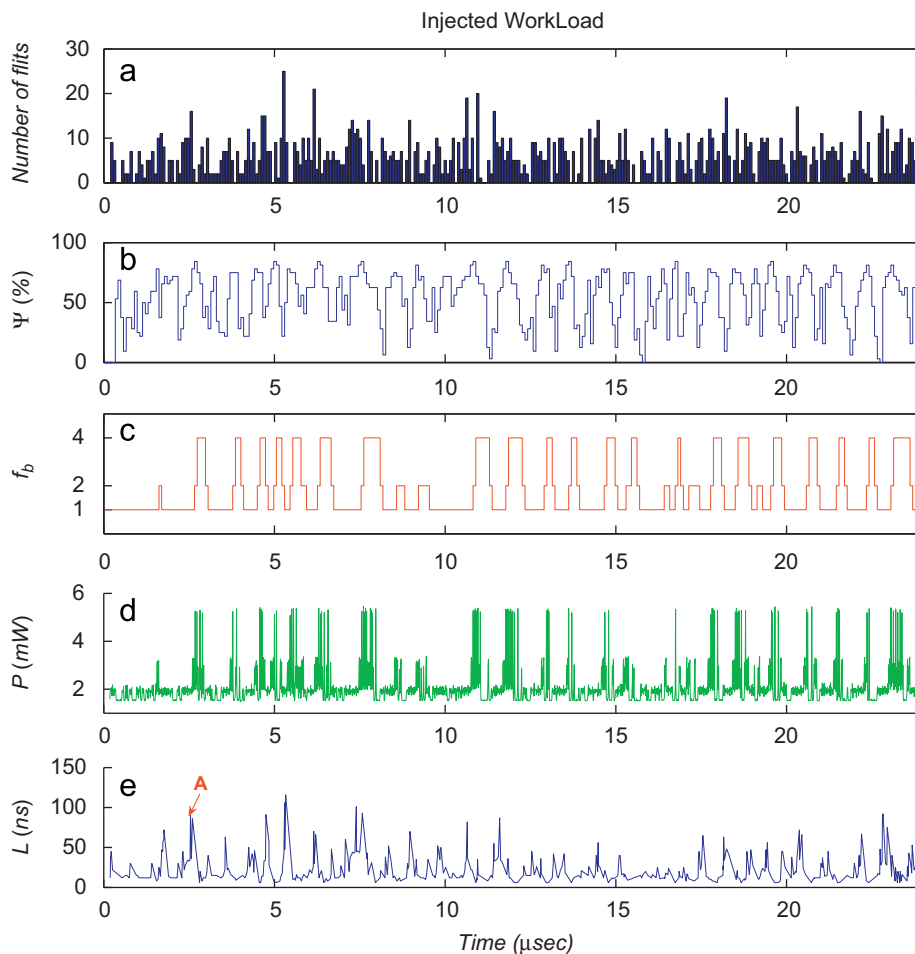


Fig. 4. Simulation result of the history-based DFS policy when T_c is eight cycles. (a) Injected workload, (b) Ψ : link utilization estimation, (c) f_b : state of boosting clock (1×, 2×, and 4×), (d) P : power consumption, and (e) L : latency for each flit.

Table 4
Characteristics of the DFS link with the workload.

Boost clock (MHz)	Average latency (ns/flit)	Peak latency (ns/flit)	End time (μs)	Dynamic power (mW)	Leakage power (mW)	Total power (mW)
100 (1×)	81.2	367	24.34	1.69	0.16	1.85
200 (2×)	15.3	97	24.06	1.96	0.16	2.12
400 (4×)	10.4	61	24.05	2.06	0.16	2.22

Table 5
Experimental results of the history-based DFS varying control period.

Control period	Average latency (ns/flit)	Peak latency (ns/flit)	End time (μs)	Dynamic power (mW)	Leakage power (mW)	Total power (mW)
8 cycles	24.3	116	24.05	1.93	0.16	2.09
16 cycles	25.2	146	24.05	1.91	0.16	2.07
32 cycles	27.6	187	24.23	1.90	0.16	2.06
64 cycles	33.2	242	24.34	1.88	0.16	2.04
128 cycles	48.21	265	24.05	1.91	0.16	2.07

dynamic power consumption for an interconnection network. Fig. 4(e) presents latency for each flit. It proves that the history-based DFS policy reduces latency at the expense of more power consumption per link. For instance, under a heavy traffic load (marked as “A”), link utilization level as well as latency increases. By applying DFS, the link controller changes boosting frequency from 1× to 4×, consequently reducing latency at the expense of more power consumption. When link utilization level goes low

because of the reduced workload, link controller decreases boosting frequency, reducing power consumption.

Table 5 summarizes the experimental results of the history-based DFS policy varying the control period from 8 to 128 cycles of the 1× clock. DFS policy enables the use of an intermediate value for power consumption between 1× and 2× clock boosting router. For instance, power consumption of DFS with eight control periods consumes 2.09 mW while 1× and 2× clock boosting

router consumes 1.85 and 2.12 mW, respectively, demonstrating the possibility of run-time power management for the given workload. Choosing a wider control period further slows down the adaptation of link frequency for the given traffic, exacerbating latency. While there is a trade-off in power and performance for the control period from 8 to 64 cycles, history-based DFS with 128 control periods consumes more power. It also increases the latency because of selecting very long control period for the given workload.

For on-chip interconnection network, the latency can be a suitable indicator to measure the performance of a network. Trade-off between power consumption and latency depends on the length of control period for the DFS policy. Even though a longer control period saves more power, it suffers from excessive latency. For the given workload, choosing control period of eight cycles is preferable for the DFS when an application requires tight timing requirements. However, a longer control period might be enough to cope with system requirements, saving more power dissipation.

In general, each application has its own power and performance demand to complete an assigned task within the desired time budget. A designer should keep in mind the system requirements in applying DFS for the on-chip interconnection network.

6. Conclusions

We have presented the notion of DFS link with fast response time using clock boosting mechanism for on-chip interconnection network. The history-based DFS policy allowed link frequency to be adjusted judiciously commensurate with the workload, balancing between power dissipation and latency penalty. The proposed DFS link and the corresponding algorithm require a simple hardware implementation, making the proposed new idea a practical option for the future NoC designs.

References

- [1] W.J. Dally, B. Towles, Route packets, not wires: on-chip interconnection networks, in: Design Automation Conference, 2001, pp. 684–689.
- [2] V. Soteriou, L.-S. Peh, Design-space exploration of power-aware on/off interconnection networks, in: ICCD'04: Proceedings of the IEEE International Conference on Computer Design, 2004, pp. 510–517.
- [3] S.E. Lee, N. Bagherzadeh, Increasing the throughput of an adaptive router in network-on-chip (NoC), in: CODES+ISSS'06: Proceedings of the 4th International Conference on Hardware/Software Codesign and System Synthesis, 2006, pp. 82–87.
- [4] V. Venkatachalam, M. Franz, Power reduction techniques for microprocessor systems, *ACM Comput. Surv.* 37 (3) (2005) 195–237.
- [5] G. Wei, J. Kim, D. Liu, S. Sidiropoulos, M.A. Horowitz, A variable-frequency parallel I/O interface with adaptive power-supply regulation, *IEEE J. Solid-State Circ.* 35 (11) (2000) 1600–1610.
- [6] J. Kim, M.A. Horowitz, Adaptive supply serial links with sub-1v operation and per-pin clock recovery, *IEEE J. Solid-State Circ.* 37 (11) (2002) 1403–1413.
- [7] L. Shang, L.-S. Peh, N.K. Jha, Dynamic voltage scaling with links for power optimization of interconnection networks, in: HPCA'03: Proceedings of the 9th International Symposium on High-Performance Computer Architecture, 2003, pp. 91–102.
- [8] F. Worm, P. lenne, P. Thiran, G.D. Micheli, An adaptive low-power transmission scheme for on-chip networks, in: ISSS'02: Proceedings of the 15th International Symposium on System Synthesis, 2002, pp. 92–100.
- [9] F. Li, G. Chen, M. Kandemir, Compiler-directed voltage scaling on communication links for reducing power consumption, in: ICCAD '05: Proceedings of the 2005 IEEE/ACM International Conference on Computer-Aided Design, IEEE Computer Society, Washington DC, USA, 2005, pp. 456–460.
- [10] D. Shin, J. Kim, Power-aware communication optimization for networks-on-chips with voltage scalable links, in: CODES + ISSS '04: Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis, IEEE Computer Society, Washington DC, USA, 2004, pp. 170–175.
- [11] V. Soteriou, N. Easley, L.-S. Peh, Software-directed power-aware interconnection networks, *ACM Trans. Archit. Code Optim.* 4 (1) (2007) 5.
- [12] X. Chen, L.-S. Peh, G.-Y. Wei, Y.-K. Huang, P. Prucnal, Exploring the design space of power-aware opto-electronic networked systems, in: HPCA '05: Proceedings of the 11th International Symposium on High-Performance Computer Architecture, IEEE Computer Society, Washington DC, USA, 2005, pp. 120–131.
- [13] E.J. Kim, K.H. Yum, G.M. Link, N. Vijaykrishnan, M. Kandemir, M.J. Irwin, et al., Energy optimization techniques in cluster interconnects, in: ISLPED '03: Proceedings of the 2003 International Symposium on Low Power Electronics and Design, ACM, NY, USA, 2003, pp. 459–464.
- [14] G. Chen, F. Li, M. Kandemir, Compiler-directed channel allocation for saving power in on-chip networks, *SIGPLAN Not.* 41 (1) (2006) 194–205.
- [15] F. Li, G. Chen, M. Kandemir, M.J. Irwin, Compiler-directed proactive power management for networks, in: CASES '05: Proceedings of the 2005 International Conference on Compilers, Architectures and Synthesis for Embedded Systems, ACM, NY, USA, 2005, pp. 137–146.
- [16] C.-L. Hsu, W.-T. Wang, Y.-F. Hong, Frequency-scaling approach for managing power consumption in NoCs, *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* E88-A (12) (2005) 3580–3583.
- [17] ARM, Arm11 mpcore (<http://www.arm.com>).
- [18] IBM, Ibm powerpc 405 embedded core (<http://www.ibm.com>).
- [19] W. Dally, B. Towles, Principles and Practices of Interconnection Networks, Morgan Kaufmann, San Francisco, 2004.
- [20] Synopsys, Synopsys design compiler, primetime px (<http://www.synopsys.com>).
- [21] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, A. Gupta, The splash-2 programs: characterization and methodological considerations, in: ISCA '95: Proceedings of the 22nd Annual International Symposium on Computer Architecture, 1995, pp. 24–36.

Seung Eun Lee received his Ph.D. degree in computer and electrical engineering from the University of California at Irvine in 2008, and his B.S. and M.S. degrees in electrical engineering and computer science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1998 and 2000, respectively. From 2000 to 2005, he was with Korea Electronics Technology Institute (KETI). Since 2009, he has been with the System Technology Lab. at Intel Corporation, Hillsboro. His research interests are in the areas of computer architecture, multiprocessor embedded systems, networks-on-chip, emerging interconnect technologies, and VLSI design.

Nader Bagherzadeh is a professor of computer engineering in the department of electrical engineering and computer science at the University of California, Irvine, where he served as a chair from 1998 to 2003. Dr. Bagherzadeh has been involved in research and development in the areas of: computer architecture, reconfigurable computing, VLSI chip design, network-on-chip, sensor networks, and computer graphics since he received his Ph.D. degree from the University of Texas at Austin in 1987.

Professor Bagherzadeh has published more than 200 articles in peer-reviewed journals and conferences. He has trained hundreds of students who have assumed key positions in software and computer systems design companies in the past 20 years. He has been a PI or Co-PI on more than 4 million dollars worth of research grants for developing next generation computer systems for applications in general purpose computing and digital signal processing.