A Voltage-Frequency Island Aware Energy Optimization Framework for Networks-on-Chip

Wooyoung Jang, *Student Member, IEEE*, and David Z. Pan, *Senior Member, IEEE*

***Abstract*—In this paper, we present a partitioning, mapping, routing and interface optimization framework for energy-efficient voltage-frequency island (VFI) based networks-on-chip. Unlike the recent work that performs tile partitioning only with voltage-frequency assignment for a given mesh network layout, our framework consists of three key VFI-aware components, i.e., VFI-aware core partitioning with voltage and frequency assignment, VFI-aware mapping, and VFI-aware routing path allocation. In addition, we develop a VFI interface and its insertion algorithm to easily satisfy performance constraints. Our methodology makes cores using the same voltage and frequency unified to single VFI. Thus, our technique considerably reduces VFI overheads such as a mixed clock first input, first output buffer and a voltage level converter up to 82% and energy consumption up to 10% compared with the state-of-the-art work. It proves that our global energy optimization framework achieves better power-performance trade-offs.**

***Index Terms*—Network-on-chip (NoC), system-on-chip (SoC), voltage-frequency island (VFI).**

I. Introduction

ACCORDING to the International Technology Roadmap for Semiconductors [1], silicon and system complexity is rocketing exponentially due to the integration of billions of transistors fueled by smaller and smaller feature sizes and more and more increasing demands for various functionalities, high performance with low design cost and short time-to-market. Consequently, current and future system-on-chip (SoC) designs are facing more and more major challenges. As SoC designs target high-performance system level integration of existing intellectual properties (IPs) such as a microprocessor, a digital signal processor, a controller, a memory and an I/O, previous dominant point-to-point interconnections and shared bus architectures, such as AMBA [2], STBus [3] and Sonics MicroNetwork [4] are becoming performance bottlenecks due to the rapid increase of communication between IPs. Networks-on-chip (NoCs) have been recently introduced as an effective solution for the scalable on-chip communication of the next generation SoC, where a shared network replaces the traditional bus structure [5]–[7]. As the better SoC platform of system integration, NoC provides more competitive features than the previous *ad hoc* global wiring mechanism.

Recently, voltage-frequency island (VFI) and globally asynchronous, locally synchronous (GALS) concepts were introduced to an NoC design for low energy consumption under performance constraints [8], where NoC tiles are partitioned into islands and each island is optimized with its own supply/threshold voltage and operating clock frequency to minimize overall energy consumption. Although the powerful energy efficiency, there are several limitations in the previous VFI-based NoC design. First, the partitioning process is only combined with voltage and frequency (VF) assignment process. Such approach limits the flexibility of VFI optimization and thus NoC energy efficiency. Second, its search for the optimal energy consumption is carried out on a hard mesh network where both communication and computation components are pre-designed. Since the network layout by a mapping process is not optimized by a VFI-aware manner, the solution space is inevitably constrained. Third, VFI-based NoC needs a good routing strategy to bring down energy consumption. It may be inefficient to insert all links between different VFIs since a mixed clock first in, first output buffer (MCFIFO) and a voltage level converter (VLC) required to interconnect different VFIs are too expensive. Therefore, pruning the links between VFIs and allocating efficient routing paths over the survived links are required to improve VFI energy efficiency, where the routing path must guarantee deadlock and livelock freeness. Last, efficient VFI interfaces are required to easily satisfy the performance constraints. If any packet generated in VFI operating with a fast clock passes VFI operating with a slow clock, it may be difficult for the packet to satisfy performance constraints. In addition, the better VFI interfaces can further reduce the number of MCFIFO and VLC required and thus NoC energy consumption.

In this paper, we propose a systematic VFI-aware energy optimization framework that consists of core partitioning with VF assignment, core mapping, routing path allocation and interfacing to tackle the aforementioned problems and further improve the energy efficiency of VFI-based NoC designs. In the proposed approach, core partitioning is carried out with VFI assignment, followed by VFI-aware mapping and VFI-aware routing path allocation. In addition, we develop the interface architecture of VFIs, which can minimize communication latency. The proposed framework provides more flexible VFI-aware NoC optimizations in terms of energy consumption under performance constraints.

The rest of this paper is organized as follows: In Section II, we present a motivational example and summarize our major

Manuscript received January 09, 2011; revised July 15, 2011; accepted August 09, 2011. Date of publication October 06, 2011; date of current version November 09, 2011. This work is supported in part by the National Science Foundation, by the Texas ARP, and by Samsung Electronics.This paper was recommended by Guest Editor E. Macii.

W. Jang was with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712 USA. He is now with Samsung Electronics, Yongin, 446-711, South Korea (e-mail: wooyoung.jang@samsung.com).

D. Z. Pan is with the Department of Electrical and Computer Engineering, University of Texas, Austin, TX 78712 USA (e-mail: dpan@ece .utexas.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JETCAS.2011.2165756

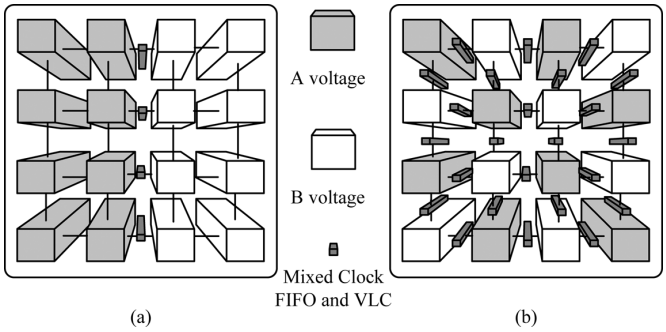


Fig. 1. Motivational NoC example.

contributions. Section III reviews related works. Section IV formulates our VFI-aware NoC optimization framework problems. Section V presents the detailed description of VFI-aware partitioning/mapping/routing/interfacing algorithms. Section VI shows experimental results compared with the state-of -the-art work [8]. Finally, Section VII is used for conclusion.

II. Motivation and Contributions

A. Motivational Example

For global energy optimization, it is highly desirable to perform core partitioning with VF assignment before mapping the cores onto NoC tiles. If a lot of MCFIFOs and VLCs required for establishing a routing path among different VFIs turn out to be too expensive and thus consume high energy, such VFI separation limits its energy efficiency.

Fig. 1, for instance, shows two VFI-based NoC designs with 16 tiles. Each tile operates at either voltage A or voltage B, depending on the computation complexity of mapped cores. After cores are mapped to tiles for the purpose of reducing hop counts and thus communication energy consumption, two different mapping results are shown in Fig. 1(a) and (b). Then, let us apply tile partitioning with VF assignment to the NoC designs, as proposed in [8]. Such approach may improve total energy consumption by running two VFIs in Fig. 1(a) since its additional design cost is only four complex routers including MCFIFO and VLC. If the energy saved by operating two VFIs is lower than the energy consumed by four complex routers, it is regarded as a desirable solution. However, in Fig. 1(b), any tile cannot operate together with other tiles at the same VF. Operating each tile as one VFI needs the complex wiring of power, ground and clock and even 24 complex routers that may be much more expensive than the energy saved by VFI separation. As a result, higher voltage of two voltages, A and B, will be used for meeting performance constraints in overall NoC such that their approach may fail to consume lower energy. This shows that tile partitioning with VF assignment alone may be misleading during NoC energy optimization. Our solution is to combine core partitioning with VF assignment, core mapping and routing path allocation together, which are considered by VFI-aware manner.

In addition, we implement various VFI interfaces and propose their insertion algorithm to minimize communication latency and further reduce energy consumption. If any packet passes VFI operating at very low clock frequency, it may be difficult for the packet to meet target communication performance. Even though a core operates at the same clock frequency as its VFI, its router and link should operate at different clock frequency satisfying performance constraints. To consider this issue, we perform the VFI-aware mapping algorithm with specific constraints and insert a pair of MCFIFO and VLC between routers or a router and a core by our VFI interface insertion algorithm.

B. Major Novelty

The main novelties and contributions of our VFI-aware optimization framework include the following.

1) We propose an NoC design methodology that is aware of VFI. After partitioning cores with VF assignment, mapping the cores and allocating routing path provides more opportunities to efficiently build unified VFIs.
2) VFI-aware mapping is performed, based on an effective region growing method. In addition, we add specific constraints for efficient VFI interface which provides short communication latency. Such VFI-aware mapping techniques fit the VFI-based NoC methodology well.
3) VFI-aware routing path allocation seeks to further reduce VFI overheads such as MCFIFO and VLC.
4) We implement various VFI interfaces and propose their insertion algorithm. Such approach achieves short communication latency and further reduces VFI overheads.
5) We show that the proposed approach makes cores running at the same voltage and clock frequency unified to single VFI with the slight increase of hop count such that it provides better energy-performance trade-offs.

III. Related Works

The thriving of NoC paradigm has triggered a burst of on-chip mapping, routing and partitioning techniques in the last decade. In [9], an energy-aware mapping was proposed for regular tile-based NoC structures. Then, it was further improved in [10] by considering packet routing flexibility during the mapping process. With on-chip communication bandwidth constraints, Murali and Micheli developed a fast shortest path algorithm for mesh-based core mapping not using integer linear programming (ILP) in [11]. With respect to routing, Class and Ni presented a deadlock-free technique called turn model for designing partially adaptive wormhole routing algorithms without virtual channels [12]. This approach was further improved by an odd-even turn model for fault-tolerant routing algorithms in [13]. A hybrid routing scheme, DyAD was proposed in [14], where a routing algorithm switched from XY to odd-even routing when congestion occurred. In [15], XY routing was extended so that congestion could be avoided along the shortest path to the destination.

There are many existing works that address the problem of VFIs generation for core-based SoC designs. The design style based on multiple VFIs was proposed in [16], where synchronous IPs in an SoC design had different voltages and frequencies. Hu *et al.* considered voltage island partitioning, assignment and floorplanning in an SoC design [17]. By using a graph-based representation, the partitioning and floorplanning steps were modeled in an integrated fashion and solved by a simulated annealing-based algorithm. Wu *et al.* considered

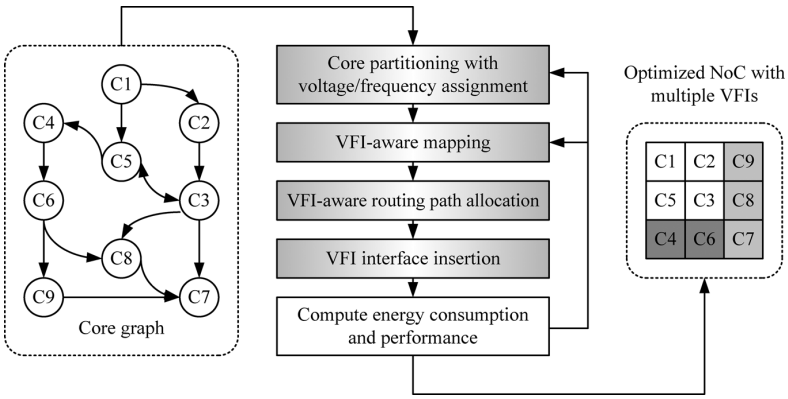


Fig. 2. Proposed VFI-aware NoC methodology.

trade-off between power and design cost under timing requirement for a VFI generation problem that was formulated as a voltage-partitioning problem and solved by a two-step heuristic algorithm [18]. In [19], the number of voltage islands determined by island partitioning was minimized after performing placement phase. Ching *et al.* considered non-slicing voltage-island partitioning to facilitate the floorplanning in [20].

As many cores have been recently interconnected by an on-chip network, the concept of the VFI design is being employed in NoC. Ogras *et al.* proposed a design methodology for partitioning NoC tiles into multiple VFIs and assigning supply/threshold voltages and corresponding clock speeds to each domain [8] and efficient techniques for on-the-fly workload monitoring and management to ensure that the system can cope with variability in the workload and various technology-related parameters [38]. Leung *et al.* proposed an NoC design with voltage islands in [21]. The approach simultaneously solved three problems, i.e., tile mapping, routing path allocation and physical voltage island generation and voltage assignment. Seiculescu *et al.* proposed a synthesis approach to obtain customized application-specific NoC that can support the shutdown of voltage islands in [22]. Liu *et al.* proposed a simultaneous task and voltage scheduling algorithm for energy minimization in NoC based designs in [23]. The energy-latency tradeoff was handled by Lagrangian relaxation.

Such a VFI-based NoC concept fits very well with a GALS design style for global on-chip asynchronous communication. The problem of selecting voltages and clock speeds for voltage/frequency islands in GALS systems was addressed in [24]. The problem of both rate and latency constrained systems was considered and a practical solution for static and application adaptive, dynamic voltage and speed scaling is provided. The field-programmable gate array (FPGA) prototype of GALS-based NoC with two synchronous IPs was presented in [25]. In [26], a method for reducing wire propagation delays in GALS-based NoC is proposed. In [8], both VFI and GALS concepts were applied to an NoC design for the minimum energy consumption.

IV. Problem Formulations

Fig. 2 shows the overall flowchart of our VFI-aware NoC optimization framework. We first partition n cores but not tiles into m VFIs, where m is given as the maximum number of allowable VFIs. Based on the result of cores partitioned, novel VFI-aware mapping and routing path allocation algorithms are performed to minimize communication energy consumption. Then, we establish unique interconnection for key traffic paths between islands to minimize the overheads of VFI. After routing path allocation is carried out, the pairs of MIFIFO and VLC are placed between routers or a core and its router. Finally, we compute its energy consumption and check whether it meets performance constraints. If the performance constraints are satisfied, an energy-efficient NoC platform with $q(\leq m)$ VFIs is obtained. Otherwise, we again perform the VFI-aware mapping with constraints described in Section V-B or all procedures with decreasing the maximum number of VFIs, m by 1.

We start to solve VFI-applied NoC issues from a core graph consisting of cores and their communication relation since a core can be one-to-one mapped onto a tile of NoC. Therefore, we assume to have an application that needs to be mapped onto NoC populated by cores as a starting point. We implement earliest deadline first (EDF) and a heuristic called energy aware scheduling (EAS) that are used for generating a core graph from a task graph [27]. The EDF approach searches any task closest deadline and then the task is the next to be scheduled for execution. On the contrary, the EAS approach is based on the idea of slack-budgeting, which allocates less slack to tasks which have a larger impact on energy consumption and performance of the application.

A. Partitioning With VF Assignment Problem

In this stage, the objective is to decide how cores are partitioned to minimize energy consumption except for communication energy consumption. We assume that the maximum number of allowable VFIs denoted by $\max\{n(\mathrm{VFI})\}$ or m, a core graph G with a set of n cores where pairs of supply voltage and threshold voltage are $(V_1, V_{t1}), (V_2, V_{t2}), \ldots, (V_n, V_{tn})$ and an NoC topology such as a mesh or a torus are given. Clock period (τ_i) for each core c_i, which can trade off with supply and threshold voltage, is defined as

$$\tau_i\left(V_i, V_{ti}\right) = \frac{K_i V_i}{\left(V_i - V_{ti}\right)^{\alpha}} \tag{1}$$

where α is a technology parameter and K_i is a design specific constant [8], [28], [29]. The operating frequency (f_j) of VFI j is determined by a core including the longest path as

$$f_j \leq \min_{i \in S_j} \left\{ \frac{1}{\tau_i (V_i, V_{ti})} \right\} \tag{2}$$

where S_j is a set of tiles that belong to VFI j. Each core can be performed with a different supply and threshold voltage and the voltage level is regarded as a legal one as long as the performance constraints are satisfied. Based on these constraints, we partition n cores into the maximum number of allowable VFIs and assign a supply and threshold voltage to each core such that total energy consumption is minimized as

$$\min \left\{ \sum_{\forall i \in G} \left(R_i C_i V_i^2 + T_i k_i V_i \exp\left(-\frac{V_t}{S_t} \right) \right) \right\} \tag{3}$$

where G is a set of n cores, R_i is the number of active cycles, C_i is total switched capacitance per cycle, T_i is the number of idle cycles, k_i is a design parameter and S_t is a technology parameter [30].

B. VFI-Aware Mapping Problem

In this section, we formulate a VFI-aware mapping problem to minimize communication energy consumption and meanwhile unify cores using the same voltage and frequency to a single island under stringent performance constraints.

Definition 1: A core graph $G'(V, E)$ partitioned in Section IV.A is a directed graph, where each vertex $v_i \in V$ represents a core and each directed edge $e_{i,j} \in E$ represents communication relation between v_i and v_j. $vol(e_{i,j})$ represents the communication volume between v_i and v_j.

Definition 2: An NoC topology graph $N(T, C)$ is a directed graph, where each vertex $t_i \in T$ represents a tile and each directed edge $c_{i,j} \in C$ represents candidates with the minimum paths from t_i to t_j. $bw(c_{i,j})$ represents the minimum bandwidth requirement from t_i to t_j.

The one-to-one mapping function $M()$ of the partitioned core V on the given NoC topology is defined as

$$M : V \rightarrow T, \ \text{s.t} \ . M(v_i) = t_j, \forall v_i \in V, \ \exists t_j \in T. \tag{4}$$

This mapping function is only defined when $n(V) \leq n(T)$, where $n(X)$ is the number of $x_i \in X$. Our mapping function has two objectives, i.e., minimizing overall communication and building a convex region with cores using the same voltage and frequency on given NoC. The near convex region selection problem with the minimum communication can be formulated as follows:

$$\min \left[L_1(N) + L_2(N - N_{\text{VFI}}) \right] \tag{5}$$

where the $L_1(N)$ and $L_2(N)$ distance are the total Manhattan distance weighted by $vol(e_{i,j})$ between t_i and t_j inside network N and the total Manhattan distance between all tiles inside network N, respectively. The objective is to find a subregion N_{VFI}.

C. VFI-Aware Routing Problem

$E_{\text{bit}}(e_{i,j})$ is the energy consumption of sending one bit of data from $M(v_i)$ to $M(v_j)$. Assuming the bit energy values are observed at V_{DD}, its energy consumption is defined as

$$E_{\text{bit}}(e_{i,j}) = \sum_{p \in L(e_{i,j})} (E_{L\text{bit}}(p) + E_{B\text{bit}}(p) + E_{S\text{bit}}(p)) \frac{V_i^2}{V_{\text{DD}}^2} \tag{6}$$

where $L(e_{i,j})$ is a set of links passed from $M(v_i)$ to $M(v_j)$ and $E_{L\text{bit}}$, $E_{B\text{bit}}$ and $E_{S\text{bit}}$ is the energy consumed by the link, buffer and switch fabric, respectively [8]. Therefore, finding a routing path from t_i to t_j is formulated as

$$\min \left[\sum_{\forall e_{i,j}} \{ vol(e_{i,j}) E_{\text{bit}}(e_{i,j}) \} + \sum_{q \leq m} \{ E_{\text{CLK}}(q) + E_{\text{VLC}}(q) + E_{\text{MCFIFO}}(q) \} \right] \tag{7}$$

where E_{CLK} is the energy overhead required to generating additional clock signals and E_{VLC} and E_{MCFIFO} are the energy overhead of VLC and MCFIFO, respectively. This formulation is subject to performance constraints expressed as

$$\frac{e_c}{f_c} + d_c \leq \text{deadline} \tag{8}$$

where e_c is the number of cycles required to complete the function of core c, and d_c is communication delay encountered when core c needs to communicate with a core mapped to a different tile.

V. VFI Optimization Framework

In this section, we present detailed algorithms for core partitioning with VF assignment, VFI-aware core mapping onto a given NoC topology, and VFI-aware routing path allocation. In addition, we show a VFI interface to easily satisfy performance constraints and further improve its energy efficiency.

A. Core Partitioning With VF Assignment

The proposed core partitioning algorithm is different from [8] that can partition tiles in a neighbor on NoC. Since our partitioning stage is performed before the stages of mapping and routing path allocation, any core unmapped to a tile can be clustered together to the same VFI. Thus, our methodology can make cores operating at the same voltage gathered as one VFI such that the number of a complex router requiring MCFIFO and VLC is minimized. Consequently, it further reduces energy consumption.

Algorithm 1 shows our core partitioning algorithm for a core graph $G(V, E)$ and the maximum number of allowable VFIs, m. Since the voltage of a core can trade off its operating frequency, the lowest voltage of each core is computed from (1) in line 1, which must satisfy performance constraint of each core. If there are k VFs used by n cores and the maximum number of accepted VFIs is m, we can choose m VF among total k VF (where $m \leq k$), where there are total ${}_kC_m$ cases. Then, the lowest VF among the chosen m VFs is assigned to each core if the performance constraint of the core is satisfied in the VF level. When the chosen m VFs are satisfied with the performance of all cores, computation energy consumption is computed from (3).

This procedure repeats all ${}_kC_m$ VF cases. After completing this procedure, we choose the best VF pair consuming the lowest energy.

Algorithm 1: Core partitioning and VF assignment

Input: $(G(V,E), \max\{n(VFI)\} = m$

1: compute the lowest voltage of each core satisfied with performance constraints using **(1)**;
2: **for** all cases that choose m VFs among k VFs used in each core **do**
3: assign the lowest operable voltage among m to all n cores;
4: **if** chosen m VFs are satisfied with performance constraints of all n cores **then**
5: compute overall energy consumption from **(3)**;
6: **end if**
7: **end for**
8: choose the best VF pair consuming minimum energy;

Output: $G'(V,E)$ partitioned into VFI

For example, there are 4 cores and the lowest voltages of each core, which satisfy their performance constraints, are 1.0, 1.2, 1.1, and 1.0 V, respectively ($k = 3$). We assume that the maximum number of allowable VFI is 2 ($m = 2$) and the operating frequency and area of all cores are same to make simple. In this example, we can choose 2 of 3 voltages, i.e., 1.0, 1.1, and 1.2 V. Thus, there are 3 cases (${}_3C_2$) we can choose: (1.0, 1.1), (1.0, 1.2), and (1.1, 1.2). Here, (1.0, 1.1) case cannot satisfy the performance constraint of the second core that must operate at least 1.2 V. On the other hand, the rest of two pairs meet performance constraints since the cores can run at 1.0, 1.2, 1.2, and 1.0 V in the second case and 1.1, 1.1, 1.2, and 1.1 V in the last case. Consequently, since the third pair consumes more energy than the second pair, (1.0, 1.2) case is chosen to compose two VFIs and assigned to all cores such that cores run at 1.0, 1.2, 1.2, and 1.0 V. Finally, a core graph $G'(V,E)$ of which the VF level is assigned is generated.

B. VFI-Aware Mapping Algorithm

Cores operating at the same VF level should be mapped to NoC tiles which build a convex region and thus, it reduces the overhead energy consumption caused by MCFIFO, VLC, and clock/power routing. We already know which cores VFIs consist of because the core partitioning with VF assignment is performed in the previous section. Therefore, this information helps selecting a region which looks as convex as possible.

In the mapping step, we use a heuristic approach using the partitioned core graph, as shown in Algorithm 2. In line 1, cores are sorted in a decreasing order by the amount of their communication and then they are mapped in the order. We define a $VF\_LIST()$ indicating whether the VF level of a core being mapped is already used on NoC. From line 3 to 11, our initial mapping algorithm starts for the sorted v_i. In line 4, the proposed mapping algorithm checks whether the VF level of a core being mapped is used throughout $VF\_LIST()$. If the VF level of the core being mapped does not exist in $VF\_LIST()$, it is mapped on any empty NoC tile with the maximum neighbor tiles or the minimum hops (line 5). Then, the VF level of the core is recorded in $VF\_LIST()$ (line 6). If the VF level of the core being mapped exists in $VF\_LIST()$, the core is mapped on any candidate tile with the same VF level (line 8). Next, additional candidates are selected in line 10, where $NSWE(t_i)$ indicates north, south, west, and east tile of the mapped t_i. The candidates are used for the next mapped cores that run at the VF level. If the core mapping is again performed due to the dissatisfaction of performance constraints, either $NS(t_i)$ or $WE(t_i)$ is used as candidate tiles. This candidate constraint makes the cores mapped to tiles in a single row or column and thus improves communication speed with our VFI interface insertion algorithm which is described in Section V-D. This procedure repeats until all cores are mapped on NoC.

Algorithm 2: VFI-Aware Mapping

Input: $G'(V,E)$, NoC topology

1: $sort(vol(v_i))$ in decreasing order;
2: $VF\_LIST() = empty$;
3: **for** all sorted v_i **do** // initial mapping
4: **if** VF level of v_i does not exists in $VF\_LIST()$ **then**
5: $M(v_i)$ on any empty tile t_j with maximum neighbors and minimum hops;
6: add VF level into $VF\_LIST()$;
7: **else then**
8: $M(v_i)$ on any candidate tile t_j with minimum hops;
9: **end if**
10: add unmapped $NSWE(t_j)$ (or $NS(t_j)$ or $WE(t_j)$) as candidate with VF of v_i;
11: **end for**
12: **for** all isolated island t_i **do** // moving of an isolated tile
13: $pair\text{-}wise\ swapping(t_i, t_j)$ to be clustered to main VFI using the same VF level under minimum traffic increase;
14: **end for**
15: **for** all t_i **do** // minimization of the overall traffics
16: $pair\text{-}wise\ swapping(t_i, t_j)$ within island for min. traffic;
17: **end for**

Output: $N(T,C)$ mapped on NoC

Our initial mapping algorithm as a near convex region solution reduces the number of an isolated tile separating from the main group of tiles (VFI) running at the same VF level. However, we cannot completely remove an isolated tile around the edge of NoC. In order to combine an isolated tile with its main VFI using the same VF level, the isolated tile is moved to the VFI if the moving cost is less than the overhead of the extra isolated tiles (line 13). Since our initial mapping does not generate an isolated tile around the center of NoC where communication is too heavy, the loss of performance and the increase of hop count caused by this pair-wise swapping of tiles in different VFIs are minimal. The procedure repeats until all isolated tiles disappear. Finally, the pair-wise swapping of tiles within each

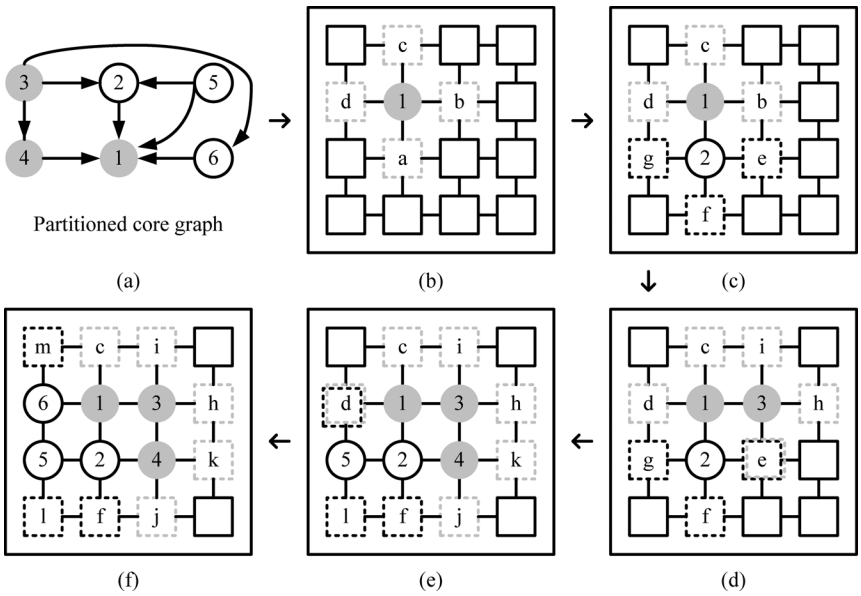


Fig. 3. Incremental core mapping on NoC.

island is executed to find the best mapping solution with the minimum hop count until it is not improved (line 16).

Fig. 3 shows the simple example of the initial mapping in Algorithm 2. In Fig. 3(a) that is the partitioned core graph, the number is the mapping order by sorting cores into the amount of communication and two groups, i.e., grey and white denoted VFI 1 and VFI 2, respectively, exist. Core 1 that has the maximum communication is placed onto the center of NoC with the maximum neighbors as shown in Fig. 3(b). Four candidates, a, b, c and d are also marked as VFI 1 for the next mapped cores using the same VF as the core 1, i.e., cores 3 and 4. Core 2 that has the next maximum communication is placed onto candidates minimizing hop count with other cores already mapped if candidates marked as VFI 2 exist. Otherwise, core 2 is only placed onto any unmapped tile that minimizes hop count with other cores already mapped. In this example, core 2 is mapped by the latter case as shown in Fig. 3(c). Three candidates, e, f, and g are also marked as VFI 2 for the next mapped core using the same VF as the core 2, i.e., core 5 and 6. Next, core 3 that has the next maximum communication is placed onto one of the VFI 1 candidates, minimizing hop count with other cores already mapped. In Fig. 3(c), there are three candidates, b, c, and d and candidate b are chosen because b generates fewer hops than c and d. Then, the three candidates, e, h and i are also marked as VFI 1, where any core operating at VFI 1 and VFI 2 can be mapped to tile e in Fig. 3(d). The procedure repeats until all cores are mapped as shown from Fig. 3(e)–(f). Since this mapping algorithm makes the region of each VFI grow toward its candidates, the VFI is prevented splitting into several VFIs using the same VF level.

C. VFI-Aware Routing Path Allocation

In this section, we present a VFI-aware routing path allocation algorithm. In [8], more than three VFIs applied to less than 25 cores could not improve overall energy consumption any more since the VFIs also generate more overheads that degrade the energy efficiency of VFI separation. The key idea of our routing path allocation is to use the minimum links between VFIs. Since NoC that is based on a mesh or torus topology has a lot of extra bandwidth, we can remove some links requiring MCFIFO and VLC if the latency of communication and the bandwidth of links satisfy performance constraints. In addition, since our VFI-aware mapping algorithm generates unified VFIs that mean any core is not split from the main group of VFI using the same voltage as the core, we use fewer MCFIFO and VLC. Consequently, the rate of energy saved by VFI separation becomes higher than the rate of energy consumed by MCFIFO and VLC as the number of VFI increases in our VFI-based NoC.

Fig. 4 shows how links between tiles are inserted briefly. After the VFI-aware mapping in Section V-B, we assume that there is no link between any tiles as shown in Fig. 4(a) including four VFIs. Next, as shown in Fig. 4(b), all links within each VFI are inserted. Then, some links between VFIs are partially inserted, as shown in Fig. 4(c). Under such an irregular NoC interconnection, routing paths allocation should minimize energy consumption and improve performance with livelock and deadlock freeness. As a result, our NoC-aware routing path allocation can achieve lower energy consumption with the tiny loss of performance and the tiny increase of hop count. In addition, communication congestion that causes the misrouting, dropping and delaying of packets is minimized.

Algorithm 3 minutely shows how links between tiles are inserted and how routing paths are allocated. Algorithm 3 consists of two parts, i.e., inserting links (lines 1 to 3) that changes an NoC topology and allocating routing paths on such an irregular interconnection (lines 4 to 12). First, we interconnect all tiles within each VFI (line 1) as shown in Fig. 4(b) because routers required neither MCFIFO nor VLC have reasonable overhead. The optimal number of the complex routers with several MCFIFOs and VLCs for connecting two islands is computed as

$$b = \left\lceil \frac{w_{i,j} vol(\mathrm{VFI}_{i,j})}{bw(\mathrm{VFI}_{i,j})} \right\rceil \tag{9}$$

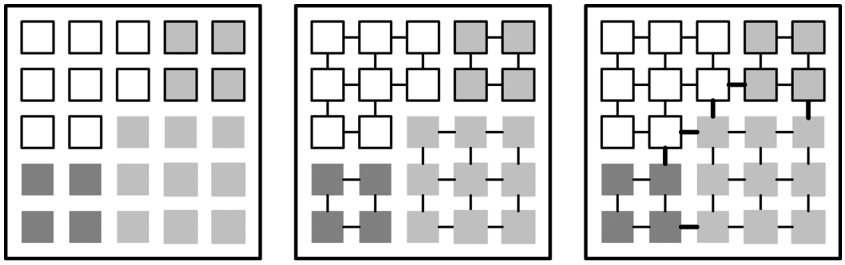


Fig. 4. Procedure of link insertion within and between VFIs. (a) No link. (b) Inserting links within VFI. (c) Inserting links between VFIs.

where $\lceil x \rceil$ is the smallest integer larger than x, $w_{i,j}$ is the weight of links between VFI i and VFI j and $vol(\mathrm{VFI}_{i,j})$ and $bw(\mathrm{VFI}_{i,j})$ are the total amount of communication volume and the minimum bandwidth requirement between VFI i and VFI j, respectively (line 2). If performance constraints are not satisfied, the weight $w_{i,j}$ increases more than 1, which means more links are inserted between adjacent VFIs. The b complex routers with MCFIFO and VLC are placed between two VFIs, where the minimum hops are generated.

Algorithm 3: VFI-Aware Routing Path Allocation

Input: $N(T, C)$

1: interconnect all tiles within each VFI;
2: compute the optimal number of routers with MCFIFO or VLS between two adjacent VFIs from **(9)**;
3: insert b routers to any place between VFIs, where the minimum hops are generated;
4: $sort(length(c_{i,j}))$ in increasing order; // Rule 1
5: **for** all $c_{i,j}$ **do**
6: bounding box including source and destination is built;
7: Dijkstra's shortest path algorithm where energy consumption of communication is computed from **(7)**; // Rule 2
8: **if** performance of **(8)** is not satisfied **then**
9: increase $w_{i,j}$ of **(9)** more than 1;
10: go to line 2;
11: **end if**
12: **end for**

Output: links insertion and deterministic, minimal and livelock/deadlock-free routing path

Here is a simple example in Fig. 5, where S1, S2, and S3 communicate with D1, D2, and D3 respectively. We assume that the amount of each communication is 1 Mbit/s, each link between tiles can contain 5 Mbit/s and $w_{i,j}$ is 1. Therefore, $vol(\mathrm{VFI}_{i,j})$ is 3 Mbit/s and $bw(\mathrm{VFI}_{i,j})$ is 5 Mbit/s such that b is equal to 1 from (9). Then, we can insert one link between two islands. Depending on the location of a link, hop counts computed by the minimum shortest path are 9, 11, and 7 Mbit/s in Fig. 5(a)–(c) respectively. Therefore, we insert one link between VFIs as shown in Fig. 5(c) because it generates the minimum hops.

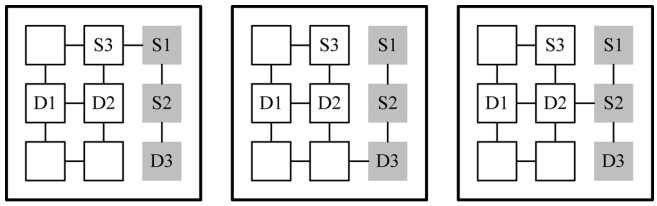


Fig. 5. Interconnection between islands.

From now, we perform routing path allocation under such an irregular NoC interconnection. In line 4 of Algorithm 3, all $c_{i,j}$ are sorted in an increasing order by their minimum hop count. For example, in Fig. 5(c), S2-to-D2 is the shortest and S1-to-D1 and S3-to-D3 have the same length. Then, we allocate routing paths based on the following two rules.

1) Rule 1: $c_{i,j}$ with few hops among C is allocated earlier to relieve communication congestion between VFIs.

In Fig. 6(a), there are two packets of which the directions are S1-to-D1 and S2-to-D2 and of which the hop counts are 2 and 6, respectively. From the *Rule 1*, the packet of which the direction is S1-to-D1 is allocated before the packet of which the direction is S2-to-D2 is allocated. The S1-to-D1 packet has only path A as the minimum shortest path whereas the S2-to-D2 packet has two paths, i.e., B and C as the minimum shortest path. If the S2-to-D2 packet is allocated to path B earlier than the S1-to-D1 packet, path A will overlap with path B since the S1-to-D1 packet has no choice. As a result, the congested packets are dropped and misrouted in a bufferless flow control mechanism like circuit switching or have long communication latency in buffered flow control mechanism like packet switching. However, if path A is allocated earlier than the S2-to-D2 packet, the path C but not the path B can be chosen as the routing path of the S2-to-D2 packet. Therefore, the routing path allocation order is important to reduce communication congestion and balance network load between VFIs. For example, in Fig. 5(c), the S2-to-D2 packet should be allocated earlier than the S1-to-D1 packet or the S3-to-D3 packet according to the rule 1.

Our routing path allocation follows *Rule 2* in the line 6 of Algorithm 3 if the VFI of packet source is different from the VFI of its destination.

2) Rule 2: If the VFI of packet source is different from the VFI of its destination, the minimum shortest routing path that passes through fewer islands is selected.

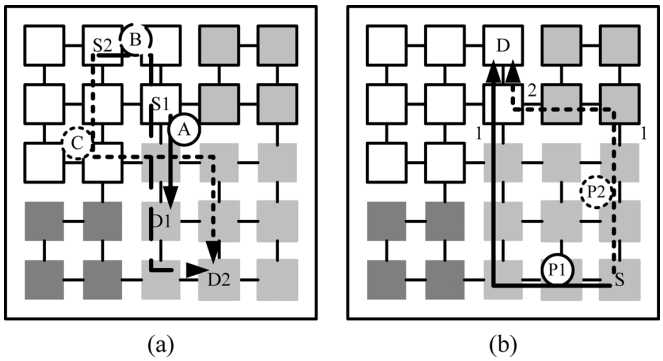


Fig. 6. Routing path allocation. (a) Rule 1. (b) Rule 2.

For example, let any packet move from S to D in Fig. 6(b). Even if several shortest routing paths can be chosen for the packet, let two routing paths, i.e., P1 and P2 considered. While both P1 and P2 are the minimum shortest paths, the routing path P1 meets one different island and the routing path P2 meets two different islands. As a result, the routing path P1 provides better performance and lower energy consumption than the routing path P2 passing two islands since the routing path P2 needs additional energy overheads.

Based on two rules, we perform our routing path allocation algorithm. For each $c_{i,j}$, a bounding box is formed (line 7) and then, the path with the minimum energy consumption is obtained within the bounding box from Dijkstra's shortest path algorithm, where its energy consumption is computed from (7). Since the routing path is allocated by a deterministic and minimal path router, both livelock and deadlock are free. If performance constrains computed from (8) is not satisfied, we increase a link weight $w_{i,j}$ between VFIs, go to line 2 and then repeat the routing path allocation. Even if there is the tiny increase of hop count in our routing path allocation due to irregular links between VFIs, the enormous energy saved by VFI separation covers such the tiny penalty.

D. VFI-Aware Interface Planning

In a VFI-based NoC design, all data and control signals are required to be converted to a different voltage by VLC and synchronized to a different clock by MCFIFO whenever they pass through a boundary between different VFIs. In the conventional VF conversion proposed in [8], MCFIFO and VLC are simply inserted between routers in different VFIs as shown in Fig. 7(a). Such a VFI interface may make it difficult to guarantee short communication latency under various VFI-based NoC design scenarios since the speed of on-chip communication depends on clock speeds used in VFIs. For example, let VFI 1 and 2 operate with 1 GHz and 100 MHz clock, respectively, in Fig. 8. Then, let any packet generated in S go to D. If the routing path of the packet is allocated to P1 which is one of the shortest path as shown in Fig. 8(a), it takes a lot of clock cycles to escape VFI 2 in terms of a VFI 1 viewpoint. The reason is that one clock cycle at VFI 2 is equal to ten clock cycles at VFI 1. Moreover, if the packet is blocked within VFI 2 due to any congestion, its latency may be severely long. This routing path is slower than another routing path P2 which is not the shortest path. Since the routing path P2 detours, it may consume more communication energy and be not free of deadlock and livelock. Actually, most microprocessors operate at several gigahertz whereas various co-processors such as peripherals, memories, specific-purposed processors and IO interface logics operate at several hundred megahertz. As described above, if any packet generated in the microprocessor operating at GHz clock speed traverses any VFI consisting of the co-processors, it is too delayed at the microprocessor's viewpoint. Even if it is a critical problem that should be solved in VFI-based NoC designs, it is not considered in the previous works yet.

We propose a new VFI interface where MCFIFO and VLC are placed between a core and its router as shown in Fig. 7(b). It is used together with the VFI interface proposed in [8]. Since the proposed interface makes the clock speed of a router choose one of two clocks, i.e., a clock used in VFI 1 and a clock used in VFI 2, the latency of packets can be greatly improved. In Fig. 8(b), three tiles in VFI 2 employ the proposed interface with MCFIFO and VLC between a core and its router and the clock speed of their routers (clock domain 3 in Fig. 7(b)) is selected to the same as that of VFI 1. Even though any packet generated in VFI 1 traverses VFI 2, its communication latency is not affected from the clock speed of VFI 2. In addition, the proposed VFI-based interface can use fewer MCFIFOs and VLCs. In Fig. 8, the number of a pair of MCFIFO and VLC in the conventional VFI interface is 16 whereas that of the pair in our VFI interface is just 6. Thus, our VFI-aware NoC interface not only reduces communication latency but also further improves VFI energy efficiency.

However, this interface may be effective when cores included in slow VFI are mapped to NoC tiles in a single row or column as shown in Fig. 8. In order to generate VFI with such a convex region, we used a different candidate selection policy in our VFI-aware mapping algorithm after mapping each core. If any VFI operates at too slow clock speed, candidates for cores in the slow VFI are just selected to either north and south tiles or east and west tiles, but not all north, south, east and west tiles. Moreover, when our VFI-aware NoC design does not meet performance constraints due to long communication latency, we repeat the VFI-aware mapping algorithm with the changed candidate selection policy, as shown in Algorithm 2 (line 10). Finally, since cores in the slow VFI are mapped only to tiles in the candidates, the VFI with a convex region can be built with tiles in a single row or column. The mapping restriction may make not only hop count slightly increase but also routers operate at high clock frequency. However, the energy efficiency degraded by such overheads cancels out the energy efficiency improved by fewer MCFIFO and VLC and even the penalty is less than the benefit of our method with a fast on-chip communication speed.

In the proposed VFI-aware NoC design, both interfaces in Fig. 7 are used together. If VFI 1 and VFI 2 inversely operate at 100 MHz and 1 GHz clock, respectively, the VFI interface in Fig. 7(a) and the VFI interface insertion in Fig. 8(a) are more desirable. Therefore, we need an efficient VFI interface insertion algorithm. As a starting point, we assume to have VFIs interconnected by the interface with MCFIFO and VLC between routers and have routers located in the upper left corner of cores. Then, we start to replace the conventional VFI interface in Fig. 7(a)

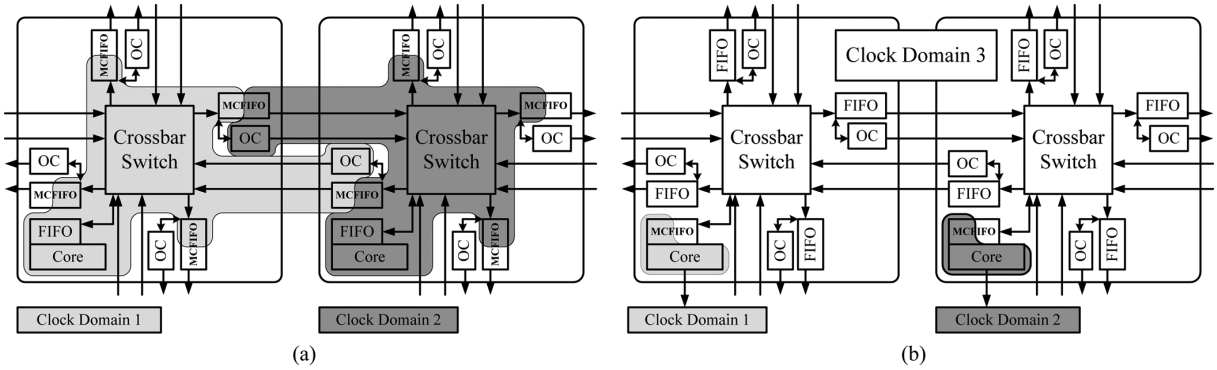


Fig. 7. VFI-based interfaces. (a) NoC tile including MCFIFO between routers [8]. (b) NoC tile including MCFIFO between core and router.

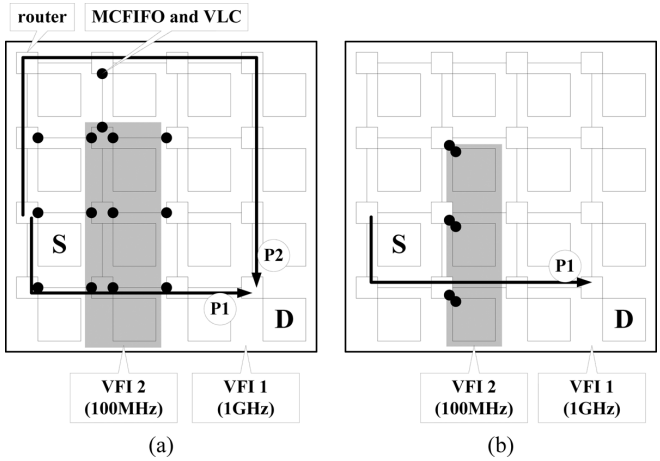


Fig. 8. MCFIFO and VLC placement planning. (a) Conventional VFI interfaces. (b) Proposed VFI interface.

to the proposed VFI interface in Fig. 7(b) from VFI operating with the lowest clock. The conventional interface in the upper or left tiles of any VFI can be replaced to the proposed interface if three conditions are satisfied as follows: 1) the upper or left tiles of the VFI must be contacted with different VFI operating at a faster clock speed, 2) the upper or left tiles of the VFI must be surrounded with both upper and lower or both left and right tiles in different VFI and 3) the interface replacement must repeat one time with a updated interface result.

Fig. 9 shows various examples of our interface replacement, where VFI colored to black operates at the slowest clock speed and VFI colored to white operates at the fastest clock speed. In Fig. 9(a), tiles A, B, C, and D satisfy conditions 1) and 2) during the first interface replacement (condition 3). Therefore, the conventional interfaces in tiles A, B, C, and D are replaced to the proposed interfaces. In Fig. 9(b), tiles A, C, and D satisfy conditions 1) and 2), whereas tile B does not satisfy condition 2) during the first interface replacement (condition 3). However, during the second interface replacement after updating the result of the first interface replacement (condition 3), tile B also satisfy condition 2). Therefore, the conventional interfaces in tiles A, B, C, and D are replaced to the proposed interfaces. In Fig. 9(c), consisting of three VFIs, the conventional interfaces of black VFI operating with the slowest clock are first replaced and then the conventional interfaces of gray VFI operating with the next slowest clock are replaced. During the first replacement, the conventional interfaces in tiles C and B are replaced to the proposed interfaces. During the second interface replacement after updating the result of the first interface replacement (condition 3), the conventional interface in tile A is also replaced to the proposed interface since tile A meets conditions 1) and 2). With such interface insertion, our VFI-aware NoC design can make communication clock speed so selectable that it is more valuable for a platform- and socket-based NoC design with VFI.

We implement the VFI-based NoC routers consisting of MCFIFO, VLC, single clock FIFO (SCFIFO), an output control (OC) and a crossbar switch as shown in Fig. 7. As described in Section V-C, crossbar switches perform deterministic and minimal path routing to minimize communication energy consumption, based on our routing path allocation rules. The OC is responsible for determining the future departure time of each packet since a physical channel must be reserved and the OC must ensure that there will be sufficient buffer spaces in the next router to store the packets. Our flow control mechanism adopts winner-take-all bandwidth allocation that allocates all of the bandwidth to one packet until it is finished or blocked before serving other packets. In addition, it can employ the advanced SDRAM- or application-aware flow control algorithms [31]–[33]. In our VFI-based NoC implementation, MCFIFO and VLC are placed between a core and its router or between routers. MCFIFO and SCFIFO are managed by wormhole flow control or virtual channel flow control mechanisms and a backpressure is used to inform the upstream nodes when they must stop transmitting packets because all of the downstream packet buffers are full. On/off flow control is adopted to avoid the loss of packets as the backpressure.

VI. Experimental Results

In this section, we show experimental results on MPEG-4 video object plane decoder (VOPD) [34] and E3S benchmark suites [35]. As the first application consists of 16 cores, the cores are one-to-one mapped to tiles on 4×4 mesh topology. The second benchmark has several applications: consumer, networking, auto-industry and telecom applications containing 12, 13, 24, and 30 tasks respectively. The benchmark also provides the information of 66 processing elements such as the size/cost of the processing elements, the maximum operating frequency, idle power consumption and task power

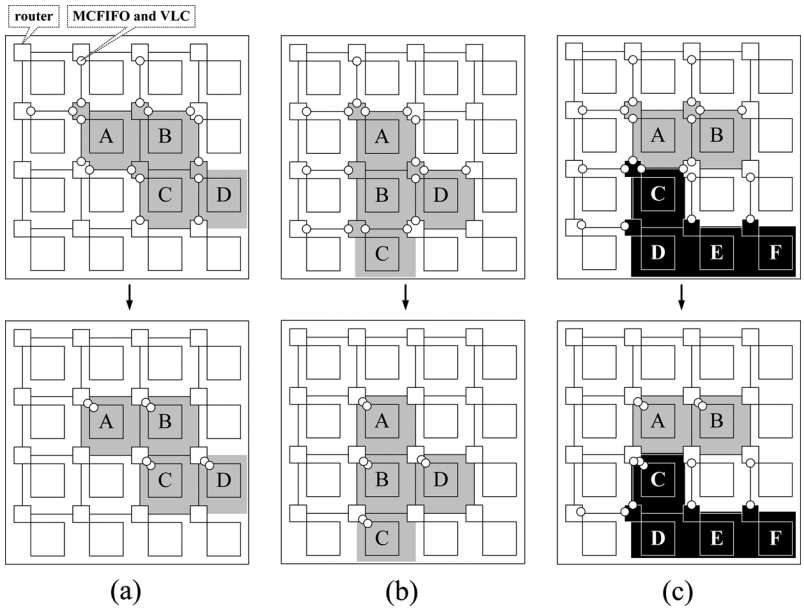


Fig. 9. Examples of our VFI interface insertion.

consumption when the tasks are performed in the processors. The tasks are scheduled onto 9, 9, 16, and 25 processors among the 66 processing elements, respectively by EDF or EAS [27] to generate a core graph. Then, they are mapped to tiles on 3×3, 3×3, 4×4, and 5×5 mesh topologies, respectively. The proposed VFI-aware NoC design is compared with the previous state-of-the-art work [8]. Since the previous work is assumed that VFI partitioning with VF assignment is performed in a hard NoC platform where communication and computation components are pre-designed, we implement NMAP [11] or A3MAP [37], the famous mapping and routing path allocation methods. We experiment our VFI-aware NoC methodology by three versions, i.e., our VFI-aware mapping algorithm combined with a conventional routing path allocation and a conventional interface, our VFI-aware mapping algorithm combined with the proposed VFI-aware routing path allocation and a conventional interface, and our VFI-aware mapping algorithm combined with the proposed routing path allocation and interface to verify the performance of mapping, routing and interface apart, denoted as VFI-M, VFI-R, and VFI-I, respectively.

Table I shows that VFI-M saves MCFIFOs and VLCs on the MPEG-4 VOPD benchmark due to our VFI-aware mapping algorithm which generates a convex region based on the VFI partitioning results. In addition, VFI-R needs the fewest MCFIFOs and VLCs. On the contrary, the VFI-aware NoC approach slightly increases hop count due to the restrictions induced by VFI-aware mapping and routing path allocation. However, the congestion is further relieved because a routing allocation order is considered for balancing network loads. The low congestion makes communication latency shorter and a communication clock speed lower.

The thorough cross-comparison of E3S benchmarks is listed in Table II. In the experiment, we assume that the changeable voltage range of cores from (1) is ±0.2 V. For example, let three cores operate at 1.2, 1.7, and 1.9 V for the minimum energy consumption. Then, if two of three voltages, i.e., 1.7 and 1.9 V are assigned for VFI-based NoC, the core operating at 1.2 V cannot be scaled into 1.7 V. The reason is that the core running at 1.2 V can be scaled up to 1.4 V by the constraint. Instead, if 1.2 and 1.9 V are assigned for VFI-based NoC, the cores can be scaled into 1.2, 1.9, and 1.9 V, respectively. Since the changeable operating range of voltage and clock frequency is small in the most real cores, this constraint is reasonable. Under this condition, tile partitioning with VF assignment by [8] generates split VFIs operating at the same VF level since task/core mapping is already performed by a VFI-unaware manner. That is, it may be difficult to include all tiles using the same voltage and frequency in a convex region. As a result, a number of MCFIFOs and VLCs are required to interconnect the separated VFIs as shown in Table II. On the contrary, our VFI-aware NoC design does not generate the split VFIs operating at the same VF level. As a result, fewer MCFIFO and VLCs are required to interconnect the VFIs. Similarly to the MPEG-4 VOPD benchmark, our VFI-aware NoC design makes hop counts increase on the E3S benchmark, yet its energy degradation can be canceled out by fewer MCFIFOs and VLCs. VFI-I requires the fewest MCFIFO and VLC, but the communication energy consumption slightly increases since some MCFIFOs and VLCs operate at higher voltage and frequency. Finally, total energy efficiency is slightly improved by

TABLE I
COMPARISON ON VIDEO OBJECT PLANE DECODER

Content	Algorithm	2 VFs	3 VFs	4 VFs
# of pair of MCFIFO and VLC	[8]	12	22	28
	VFI-M	10	14	20
	VFI-R	2	4	6
Hop count	[8]	4309	4309	4309
	VFI-R	4353	4211	4211
Congestion (MB/s)	[8]	923	923	923
	VFI-R	516	613	613

TABLE II
CROSS COMPARISON ON E3S BENCHMARK

Content	Algorithm	Telecommunication				Auto-Industry			Networking			Consumer		
		2 VFs	3 VFs	4 VFs	5 VFs	2 VFs	3 VFs	4 VFs	2 VFs	3 VFs	4 VFs	2 VFs	3 VFs	4 VFs
# of pair of MCFIFO and VLC	[8]	40	48	58	58	22	24	30	16	16	18	8	16	20
	VFI_M	26	28	44	40	16	20	26	12	12	14	6	10	14
	VFI_ R	20	22	26	32	12	14	22	10	10	12	6	8	12
	VFI_ I	14	20	24	32	8	12	20	8	8	10	6	6	12
Hop count	[8]	107	107	107	107	172	172	172	79692	79692	79692	30	30	30
	VFI_ R	133	133	138	153	178	193	205	83886	83886	109051	33	35	39

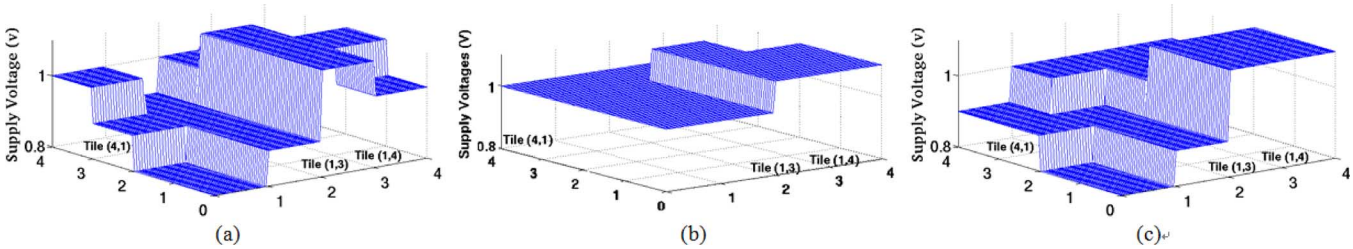


Fig. 10. VFI partitioning illustration on 4×4 NoC. (a) VF map after task/core mapping. (b) NoC partitioning by [8]. (c) Proposed VFI-aware NoC.

our VFI interface approach. In addition, VFI-I improves 27% communication latency on average.

Fig. 10 illustrates the visual comparison of [8] and our VFI-aware approach which are performed on 4×4 mesh topology for the MPEG-4 VOPD application. Fig. 10(a) is the result of core/task mapping by NMAP [11], where the voltage of tiles is assigned to the lowest voltage which cores require to execute the scheduled tasks. Since cores are not mapped by VFI-aware manner, cores using the same VF level are split over NoC as shown in Fig. 10(a). Since cores operating at 1 V are split to three VFIs, a VF map with six VFIs is shown even if 4 VF levels are used. Based on the VF map, VFI partitioning with VF assignment is performed by [8] and its result is shown in Fig. 10(b). The VFI partitioning achieves the best energy consumption when two VFIs are built. However, some cores operate at a higher voltage than any voltage which the cores require to satisfy performance constraints. Therefore, it is a suboptimal solution since task/core mapping are already performed by VFI-unaware manner even if the VFI partitioning performance considerably depends on the mapping stage. On the other hand, our VFI-aware approach consisting of core partitioning with VF assignment, VFI-aware mapping and VFI-aware routing path allocation clusters tiles using the same VF level to single VFI such that it clearly provides better partitioned VFI as shown in Fig. 10(c). As a result, all tiles operate at the optimal voltage and frequency they require to satisfy performance constraints and thus it is beneficial for low energy consumption as well as the global routing of power and clock.

Table III shows that our VFI-aware NoC design consumes less energy than [8]. Our approach makes all cores operate at the own voltage and frequency and needs fewer MCFIFOs and VLS since all cores running at the same VF level can be clustered into single VFI. As a result, our VFI-aware NoC optimization further saves energy consumption as the number of VFI increases. On the other hands, the previous state-of-the-art approach [8] can improve energy consumption only when the number of built VFI is 2 or 3. For the network application, since hop count enormously increases in case that the number of built VFI is 4, our approach consumes more energy than [8] as shown in Table III. Our approach chooses the lowest of all possible cases. That is, since our energy consumption in 3 VFIs is the lowest, 3 VFIs is finally selected, which is also lower than the lowest energy consumption of [8]. Finally, the runtime of our VFI-aware NoC optimization ranges from a few microseconds to a few seconds, where the complexities of partitioning, mapping, and routing path allocation, interface planning are $O((k-m)!/m!)$, $O(V \log V)$, $O(|E| + |V| \log |V|)$, and $O(V)$.

TABLE III
NOC ENERGY CONSUMPTION COMPARISON.

Benchmark	Algorithm	Normalized Total Energy Consumption			
		1 VFI	2 VFIs	3 VFIs	4 VFIs
Consumer	[8]	1	0.56	0.53	0.54
	VFI_ R	1	0.55	0.51	0.50
	VFI_ I	1	0.55	0.50	0.50
Network	[8]	1	0.8	0.79	0.79
	VFI_ R	1	0.78	0.76	0.89
	VFI_ I	1	0.77	0.76	0.89
Auto-industry	[8]	1	0.69	0.65	0.67
	VFI_ R	1	0.63	0.59	0.58
	VFI_ I	1	0.61	0.58	0.57
Telecom	[8]	1	0.58	0.57	0.58
	VFI_ R	1	0.53	0.51	0.49
	VFI_ I	1	0.50	0.50	0.48

VII. CONCLUSION

In this paper, we proposed a systematic energy optimization framework, including core partitioning with VF assignment, VFI-aware mapping, VFI-aware routing and VFI-aware interface insertion for VFI-based NoC designs. Our VFI-aware NoC design makes tiles mapped cores using the same voltage

and frequency level clustered to single VFI. In addition, our VFI interface further improves energy consumption with fast on-chip communication. Consequently, our VFI-aware NoC optimization framework reduces VFI design cost that degrades energy efficiency by VFI overheads. Compared to the recent state-of-the-art NoC design technique with VFI [8], our VFI-aware optimization framework demonstrates an energy efficiency improvement of 10% and the overhead reduction of 82% under a variety of system constraints.

REFERENCES

[1] ITRS, "International technology roadmap for semiconductors," 2009.

[2] *"AMBA Open Specifications,"* ARM, San Jose, CA [Online]. Available: http://www.arm.com

[3] *"STBus Communication System Concepts and Definitions,"* STMicroelectronics, [Online]. Available: http://www.st.com

[4] D. Wingard, "Micronetwork-based integration for SoCs," in *Proc. Des. Autom. Conf.*, 2001, pp. 673–677.

[5] L. Benini and G. De Micheli, "Network on chips: A new SoC paradigm," *Computer*, vol. 35, no. 1, pp. 70–78, Jan. 2002.

[6] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proc. Des. Autom. Conf.*, 2001, pp. 684–689.

[7] R. Marculescu, U. Y. Ogras, L.-S. Peh, N. E. Jerger, and Y. Hoskote, "Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives," *IEEE Trans. Comput.-Aided Des. Integr. Curcuits Syst.*, vol. 28, no. 1, pp. 3–21, Jan. 2009.

[8] U. Y. Ogras, R. Marcuescu, P. Choudhary, and D. Marculescu, "Voltage-frequency island partitioning for GALS-based networks-on-chip," in *Proc. Des. Autom. Conf.*, 2007, pp. 110–115.

[9] J. Hu and R. Marculescu, "Energy-aware mapping for tile-based NoC architectures under performance constraints," in *Proc. Asian and South Pacific Des. Autom. Conf.*, 2003, pp. 233–239.

[10] J. Hu and R. Marculescu, "Exploiting the routing flexibility for energy/performance aware mapping of regular NoC architectures," in *Proc. Design, Autom. & Test in Eur.*, 2003.

[11] S. Murali and G. De Micheli, "Bandwidth-constrained mapping of cores onto NoC architecture," in *Proc. Design, Autom. & Test in Eur.*, 2004, pp. 896–901.

[12] C. J. Class and L. M. Ni, "The turn model for adaptive routing," *J. ACM*, vol. 41, no. 5, pp. 874–902, Sep. 1994.

[13] G.-M. Chiu, "The odd-even turn model for adaptive routing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, no. 7, pp. 729–738, Jul. 2000.

[14] J. Hu and R. Marculescu, "DyAD – Smart routing for networks-on-chip," in *Proc. Des. Autom. Conf.*, 2004, pp. 260–263.

[15] M. Li, Q.-A. Zeng, and W.-B. Jone, "DyXY—A proximity congestion-aware deadlock-free dynamic routing method for network-on-chip," in *Proc. Des. Autom. Conf.*, 2006, pp. 849–852.

[16] D. E. Lackey, P. S. Zuchowski, T. R. Bednar, D. W. Stout, S. W. Gould, and J. M. Cohn, "Managing power and performance for system-on-chip designs using voltage islands," in *Proc. Int. Conf. Comput.-Aided Des.*, 2002, pp. 195–202.

[17] J. Hu, Y. Shin, N. Dhanwada, and R. Marculescu, "Architecting voltage islands in core-based system-on-a-chip designs," in *Proc. Int. Symp. on Low Power Electron. Des.*, 2004, pp. 180–185.

[18] H. Wu, I.-M. Liu, M. D. F. Wong, and Y. Wang, "Post-placement voltage island generation under performance requirement," in *Proc. Int. Conf. Comput.-Aided Des.*, 2005, pp. 309–316.

[19] H. Wu, M. D. F. Wong, and I.-M. Liu, "Timing-constrainted and voltage-island-aware voltage assignment," in *Proc. Des. Autom. Conf.*, 2006, pp. 429–432.

[20] R. L. S. Ching, E. F. Y. Young, K. C. K. Leung, and C. Chu, "Post-placement voltage island generation," in *Proc. Int. Conf. Comput.-Aided Des.*, 2006, pp. 641–646.

[21] L.-F. Leung and C.-Y. Tsui, "Energy-aware synthesis of networks-on-chip implemented with voltage island," in *Proc. Des. Autom. Conf.*, 2007, pp. 128–131.

[22] C. Seiculescu, S. Murali, L. Benini, and G. De Micheli, "NoC topology synthesis for supporting shutdown of voltage island in SoCs," in *Proc. Des. Autom. Conf.*, 2009, pp. 822–825.

[23] Y. Liu, Y. Yang, and J. Hu, "Clustering-based simultaneous task and voltage scheduling for NoC systems," in *Proc. Int. Conf. Comput.-Aided Des.*, 2010, pp. 277–283.

[24] K. Niyogi and D. Marculescu, "Speed and voltage selection for GALS systems based on voltage/frequency islands," in *Proc. Asian and South Pacific Des. Autom. Conf.*, 2005, pp. 292–297.

[25] J. Quartana, S. Renane, A. Baixas, L. Fesquet, and M. Renaudin, "GALS systems prototyping using multiclock FPGAs and asynchronous network-on-chips," in *Proc. International Conference on Field Programmable Logic and Applications*, Aug. 2005, pp. 299–304.

[26] G. Campobello *et al.*, "GALS networks on chip: A new solution for asynchronous delay-insensitive links," in *Proc. Des., Autom. Test in Eur.*, Mar. 2006, pp. 1–6.

[27] J. Hu and R. Marculescu, "Communication and task scheduling of application-specific networks-on-chips," *IEE Proc. Comuters Digital Techn.*, pp. 643–651, Sep. 2006.

[28] S. M. Martin, K. Flautner, T. Mudge, and D. Blaauw, "Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads," in *Proc. Int. Conf. Comput.-Aided Des.*, 2002, pp. 721–725.

[29] T. Sakurai and A. R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE J. Solid-State Circuits*, vol. 25, no. 2, pp. 584–594, Apr. 1990.

[30] J. A. Butts and G. S. Sohi, "A static power model for architects," in *Proc. Int. Symp. Microarchitecture*, Dec. 2000, pp. 191–201.

[31] W. Jang and D. Z. Pan, "An SDRAM-aware router for networks-on-chip," in *Proc. Des. Autom. Conf.*, 2009, pp. 800–805.

[32] W. Jang and D. Z. Pan, "Application-aware NoC design for efficient SDRAM access," in *Proc. Des. Autom. Conf.*, 2010, pp. 453–456.

[33] W. Jang and D. Z. Pan, "An SDRAM-aware router for networks-on-chip," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 29, no. 10, pp. 1572–1585, Oct. 2010.

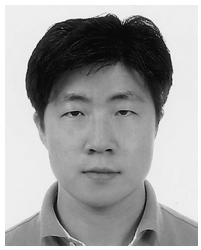
[34] E. B. Van der Tol and E. G. T. Jaspers, "Mapping of MPEG-4 decoding on flexible architecture platform," in *SPIE 2002*, Jan. 2002, pp. 1–13.

[35] R. P. Dick, Embedded System Synthesis Benchmarks Suites (E3S) [Online]. Available: http://www.ece.northwestern.edu/~dickrp/e3s/

[36] W. Jang and D. Z. Pan, "A3MAP: Architecture-aware analytic mapping for networks-on-chip," in *Proc. Asian and South Pacific Des. Autom. Conf.*, Jan. 2010, pp. 523–528.

[37] W. Jang, D. Ding, and D. Z. Pan, "A voltage-freqency island aware energy optimization framework for networks-on-chip," in *Proc. Int. Conf. Comput.-Aided Des.*, Nov. 2008, pp. 264–269.

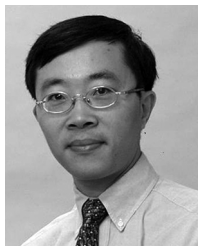
[38] U. Y. Ogras, R. Marculescu, D. marculescu, and E. G. Jung, "Design and management of voltage-frequency island partitioned networks-on-chip," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 17, no. 3, pp. 330–341, Mar. 2009.



Wooyoung Jang (S'08) received the B.E. degree in radio science and technology from Kyung Hee University, Suwon, South Korea, in 1998, the M.S. degree in electrical and computer engineering from Yonsei University, Seoul, South Korea, in 2000 and the Ph.D. degree in electrical and computer engineering from the University of Texas at Austin, in 2011.

Since 2000, he has been with System Large Scale Integration Division, Samsung Electronics, Suwon, South Korea as a Senior Engineer. His current research interests include computer architecture and nanometer physical design for on-chip communication.

Mr. Jang was the recipient of the SK telecom Scholarship for 1994–1997, the Samsung Outstanding Achievement Award in 2005, and the Samsung Scholarship for 2006–2011.



David Z. Pan (S'97–M'00–SM'06) received the Ph.D. degree (with hons.) in computer science from the University of California at Los Angeles (UCLA), Los Angeles, in 2000.

From 2000 to 2003, he was a Research Staff Member with the IBM T. J. Watson Research Center, Yorktown Heights, NY. He is currently an Associate Professor (with tenure) with the Department of Electrical and Computer Engineering, the University of Texas at Austin. He has published over 140 technical papers in refereed journals and conferences, and is the holder of six U.S. patents. His research interests include

nanometer physical design, design for manufacturing, vertical integration of technology/CAD/architecture, and design/CAD for emerging technologies.

Dr. Pan has served as an Associate Editor for four premier IEEE journals, including IEEE TRANSACTIONS ON COMPUTER AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS (TCAD), IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATED (VLSI) SYSTEMS, AIEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS (2006–2007). He has also served as an Area Editor for the *Journal of Computer Science and Technology*, an Associate Editor for the *IEEE CAS Society Newsletter* and a Guest Editor of TCAD Special Section on "International Symposium on Physical Design" in 2007 and 2008. He is a member in the Design Technology Working Group of International Technology Roadmap for Semiconductor (ITRS). He has served as the Chair of the IEEE CANDE Committee and the ACM/SIGDA Technical Committee on Physical Design. He is the General Chair of ISPD 2008 and Steering Committee Chair of ISPD 2009. He has served in the Technical Program Committees of major VLSI/CAD conferences, including ASPDAC (Subcommittee Chair), DAC (Subcommittee Chair), DATE, ICCAD (Subcommittee Chair), ISPD (Program Chair), ISQED (Topic Chair), ISCAS (CAD Track Chair), among many others. He has served as a Technical Advisory Board member of Pyxis Technology, Inc. He has received a number of awards for his research contributions and professional services, including the ACM/SIGDA Outstanding New Faculty Award (2005), NSF CAREER Award (2007), UCLA Engineering Distinguished Young Alumnus Award (2009), SRC Inventor Recognition Award three times (2000 and 2008), IBM Faculty Award four times (2004–2006, 2010), DATE Best IP Award (2009), ASPDAC Best Paper Award (2010), ISPD Best Paper Award (2011), SRC Techcon Best Paper in Session Award twice (1998 and 2007), ICICDT Best Student Paper Award (2009), Dimitris Chorafas Foundation Research Award (2000), eASIC Placement Contest Grand Prize (2009), ISPD Routing Contest Awards (2007), and ACM Recognition of Service Award (2007 and 2008). He is an IEEE CAS Society Distinguished Lecturer for 2008–2009. Dr. Pan is a Senior Member of IEEE, and a member of ACM/SIGDA, SPIE, and ASEE.