Des Autom Embed Syst (2011) 15:89–109
DOI 10.1007/s10617-011-9070-x

Communication-aware VFI partitioning for GALS-based networks-on-chip

Dongkun Shin · Woojoong Kim · Soontae Kwon · Tae Hee Han

Received: 4 May 2010 / Accepted: 18 February 2011 / Published online: 10 March 2011

Abstract The voltage/frequency island (VFI) design paradigm is a practical architecture for energy-efficient networks-on-chip (NoC) systems. In VFI-based NoC systems, each island can be operated with different voltage and clock frequency and thus it is important to carefully partition processing elements (PEs) into islands based on their workloads and communications. In this paper, we propose an energy-efficient design scheme that optimizes energy consumption and hardware costs in VFI-based NoC systems. Since on-chip networks take up a substantial portion of system power budget in NoC-based systems, the proposed scheme uses communication-aware VFI partitioning and tile mapping/routing algorithms to minimize the inter-VFI communications. Experimental results show that the proposed design technique can reduce communication energy consumption by 32–51% over existing techniques and total energy consumption by 3–14%.

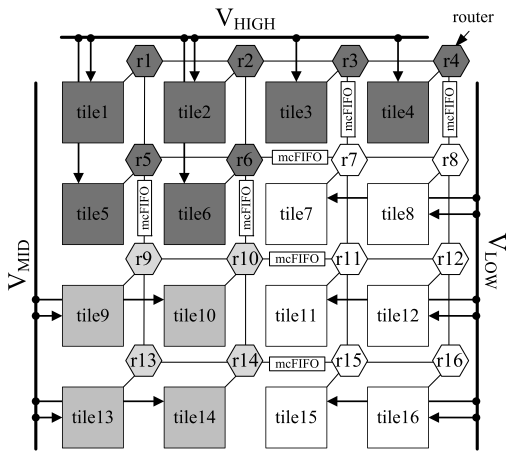
Keywords Networks-on-chip · Voltage/frequency island · Low-power design · GALS · Interconnection network

1 Introduction

Recently, the use of multiprocessor system-on-chip (MP-SoC) platforms has emerged as an important integrated circuit design trend for high-performance computing applications. As the number of processors on such platforms continues to increase, monolithic bus-based interconnect architectures will not be able to support the increased complexity. To support higher degrees of integration, networks-on-chip (NoC) [1, 2] is seen as an efficient on-chip communication infrastructure, where the interconnection network replaces the traditional shared bus structure. The NoC design paradigm enables the integration of an exceedingly high number of computational and storage blocks in a single chip by overcoming complex

D. Shin · W. Kim · S. Kwon · T.H. Han (✉)
School of Information and Communication Engineering, Sungkyunkwan University,
300 Cheoncheon-dong, Jangan-gu, Suwon 440-746, South Korea
e-mail: than@skku.edu

Fig. 1 An example of VFI-based NoC



on-chip communication problems with a more structured and modular network interface and by allowing scalable designs.

An NoC-based system is typically divided into regular tiles, where each tile is either a processing element or a storage element. Instead of dedicated wires, each tile is connected to an interconnection network that routes packets between tiles. The router consists of input and output links, buffers and a crossbar switch.

To minimize energy consumption, each tile can be assigned an optimal voltage/frequency in consideration of both workload and performance constraints. In order to use a different clock frequency for each tile, it would be appropriate to implement the NoC by the globally asynchronous, locally synchronous (GALS) design paradigm [3, 4], where locally synchronous blocks communicate with one another asynchronously.

However, optimized unique voltage/frequency assignment on each tile requires a significant number of mixed-clock FIFOs (mcFIFOs) [5] and voltage level converters (VLCs), which invoke performance/energy penalties, design complexities and silicon area costs. To alleviate these problems, a VFI-based NoC design paradigm has recently been introduced [6–8], where an NoC is partitioned into several voltage/frequency islands (VFIs). Each island is composed of multiple tiles and is optimized with its own supply voltage and operating frequency to minimize overall energy consumption. Figure 1 shows an example of NoC consisting of three VFIs which use voltages V_{HIGH}, V_{MID} and V_{LOW}, respectively. Seven mcFIFOs are required to connect the routers which use different voltages/frequencies. The use of VFIs in the NoC design can reduce the implementation overhead of single voltage, single clock frequency NoC design and can exploit energy-performance tradeoffs more flexibly.

As a recent example, Intel announced the 24 tiled "single-chip cloud computer" [9], where four adjacent tiles are grouped into one voltage island and each tile can run at a different frequency. Software can dynamically configure voltage and frequency of each tile to attain power consumptions from 125 W to as low as 25 W at run time.

In designing VFI-based NoCs, an important design decision is how to partition several processing elements into VFIs. Since all processing elements in a VFI should use the same voltage and clock frequency, processing elements which demand similar voltages and clock frequencies should be grouped into a single VFI. In addition, it is also important to minimize communications between VFIs since inter-VFI communication requires mcFIFOs.

As with other multiprocessor-based systems, the design flow of NoC-based systems involves several interacting steps. In a typical multiprocessor system, the design flow includes two key steps: *task assignment* and *task scheduling*. Given a task graph with design constraints (e.g., execution time and power consumption) and processing elements (PEs), each task is first assigned to an appropriate PE (*task assignment*). Then, each task is scheduled for execution within the PE (*task scheduling*). However, in NoC-based systems, two additional steps are necessary: *tile mapping* and *routing path allocation*. The tile mapping step maps a PE to one of the tiles in an NoC platform. The routing path allocation step determines communication paths between tiles. We call these two steps together *network assignment*. For VFI-based NoCs, in addition to these steps, tiles should be grouped into several VFIs (*VFI partitioning*) and each VFI should be assigned a voltage and clock frequency (*V/F assignment*).

In this paper, we propose an overall VFI-aware energy optimization framework for NoCs. The proposed framework includes VFI partitioning and V/F assignment as well as tile mapping and routing. It is much more energy-efficient by considering inter-VFI communications in all design steps. Experimental results show that the proposed design technique can reduce communication energy consumption by 32–51% over existing techniques and total energy consumption by 3–14%.

The rest of the paper is organized as follows. In Sect. 2, we briefly review the related works on energy-efficient NoC design techniques. The problem formulation and overall design flow are presented in Sect. 3. The detailed design techniques are described in Sect. 4. We present experimental results in Sect. 5. Section 6 concludes with a summary and directions for future work.

2 Related works

Several research groups have investigated system-level techniques for minimizing energy consumption in NoC-based systems. The first technique is to dynamically adjust the operating voltage and clock frequency of a communication link. Shang et al. [10] developed a history-based dynamic voltage scaling (DVS) policy based on the utilization of the link and the input buffer. Worm et al. [11] proposed an adaptive low-power transmission scheme which minimizes the energy required for reliable communications, while satisfying a QoS constraint by dynamically controlling the voltage on the links.

The second technique is a dynamic link shutdown (DLS) [12] which powers down links dynamically when their utilizations are below a certain threshold level. The DLS technique requires an adaptive routing strategy that intelligently uses a subset of links for minimizing energy consumption. Soteriou and Peh [13] proposed a proactive dynamic link shutdown technique for communication links, which turns links on or off depending network statistics such as input buffer utilization.

The third technique is to minimize the communication traffic by communication-aware application mapping. Hu and Marculescu [14] proposed a design technique which reduces the communication traffic by optimizing the communication distance between two PEs with high communication demands at tile mapping and routing steps. Shin and Kim [15] proposed a genetic algorithm-based NoC design framework which optimizes the voltage/frequency of communication links as well as the communication distance. Marcon et al. [16] considered communication dependencies as well as computation dependencies in communication scheduling.

There are also several compiler-directed approaches for energy-efficient NoCs. Chen et al. [17] introduced a compiler-level technique, 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. [18] proposed a compiler-driven approach where a compiler analyzes the application code and extracts communication patterns among parallel processors. Based on these patterns, the compiler decides the optimal voltage/frequency for communication links. Kandemir and Ozturk [19] presented a compiler-directed voltage/frequency scaling algorithm for both the PEs and communication links. They use a critical path analysis technique to estimate slack times and an integer linear programming (ILP) to determine the voltage/frequency of each PE and link.

While these works focus on reducing communication costs to optimize energy consumption, VFI-based NoC focuses mainly on reducing the energy consumption of processing elements. Various GALS-based NoC architectures that focus on link level synchronization techniques in a multi-clock domain SoC have been introduced, such as DSPIN [20], ASPIN [20], MANGO [21] and ANOC [22].

For VFI-based NoCs, chip partitioning into multiple VFIs is a critical issue. Ogras et al. [6, 7] first proposed the VFI partitioning and static voltage assignment methodology taking into account the overhead due to mcFIFO and VLC. However, they did not consider the tile mapping and routing path allocation problem. Since PEs of a VFI should be adjacent in NoC platform to each other, the VFI partitioning should be combined with tile mapping and routing path allocation. Otherwise, a VFI partitioning alone may mislead NoC energy optimization.

Leung and Tsui [23] grouped PEs on the critical path in task graph to form a single voltage island. They tried to minimize PE energy consumptions only and did not consider the communication overhead between VFIs. In addition, they assumed that each PE has been assigned only one task to ease the critical path analysis. However, in real implementations, one PE can execute multiple tasks at different time slices.

Jang et al. [8] presented a VFI-aware energy optimization framework that includes VFI partitioning, tile mapping and routing path allocation. Their VFI partitioning algorithm searches the full design space to find the optimal partition solution and thus may require significant search time when the complexity of the VFI-based NoC is large. Moreover, the proposed VFI partitioning does not consider the communication costs of the partition solution. Nevertheless, they assumed that each task assigned to a tile has its own deadline to determine the voltage/frequency of each tile, and so this may not be a practical solution.

Guang et al. [24] proposed an autonomous DVFS technique for VFI-based NoC. The local DVFS monitor adjusts the voltage and frequency of each island based on the network load at run time.

Ghosh et al. [25] proposed a mixed integer linear program (MILP) formulation of the VFI-based NoC design problem. They handled task assignment, tile mapping, routing, and voltage/frequency assignment problems. To model with a MILP formulation, they assume each PE has a candidate set of discrete voltages/frequencies. The MILP requires an excessive time to find optimal solution especially when the number of tiles in NoC is large. In addition, the technique does not cover the VFI partitioning problem. Instead, it optimizes only the energy overhead for communications between different PEs operating at different voltages for a given maximum number of voltage islands. Such a technique can neither consider the isolation of VFIs nor simplify the power delivery network, which is handled by our proposed technique.

Our algorithm has similar design flows in [8] and adopts their ideas for tile mapping and routing. However, we use a more practical task model and propose a communication-aware

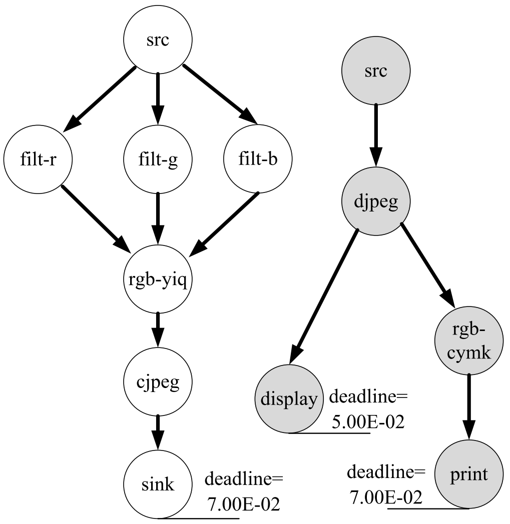


Fig. 2 An example of a task graph $G(V, E)$ (consumer-mocsyn.tgff in E3S [28])

VFI partitioning heuristic. In addition, we improve the tile mapping and routing techniques by considering communication costs. The differentiation points are explained in detail in Sect. 3.3.

3 VFI-based NoC design

3.1 Specification and architectural model

We represent a periodic real-time application by a task graph (TG) $G = (V, E)$ as shown in Fig. 2, which is a directed acyclic graph. V is the set of tasks and E is the set of directed edges between tasks. We assume that the processing element for a task τ_i in V is given as an input and is denoted by $r(\tau_i)$. One PE can be assigned multiple tasks. Each task τ_i is associated with its worst-case execution cycles at PE $r(\tau_i)$, $w(\tau_i)$. We design an NoC-based system by using the worst-case workload of each task. It is assumed that the worst-case execution cycles of each task are known. This assumption is common at most design techniques [8, 23, 26]. The worst-case execution cycles can be estimated by a timing analysis tool such as [27] or a profiling tool.

The set of PEs is represented by R. In a TG, each directed edge $e(\tau_i, \tau_j)$ represents a precedence relation between τ_i and τ_j. That is, $e(\tau_i, \tau_j)$ means that task τ_i must complete its execution before task τ_j starts its execution. (For descriptive purposes, we will denote $e(\tau_i, \tau_j)$ by $e_{i,j}$.) A task τ_i may have a deadline d_i, which must be met to ensure correct functionality of the application. If a task is not a sink node in the graph, it may have no deadline. But, the sink node must have a deadline. Each edge $e_{i,j}$ is associated with a value $w(e_{i,j})$ which indicates the amount of communication data required between τ_i and τ_j when τ_i and τ_j are allocated to different PEs.

We denote a VFI-based NoC-based system N with $n \times n$ tiles and m islands as $\langle T, \Pi \rangle$, where $T = \{t_1, \dots, t_n, \dots, t_{n^2}\}$ is the set of tiles, and $\Pi = \{I_1, \dots, I_m\}$ is the set of voltage/frequency islands. The number of VFIs, m, determines the cost of VLCs. We assume a regular mesh communication architecture as shown in Fig. 1. We will denote the link between t_i and t_j by $\ell_{i \to j}$. For a link ℓ_i, $W(\ell_i)$ indicates the total amount of data transferred across the link. $W(\ell_i)$ should not be larger than the bandwidth constraint of a link BW_{max}.

3.2 Problem formulation

For a given task graph $G = \langle V, E\rangle$ and a VFI-based NoC architecture $\langle T, \Pi\rangle$, the required design decisions are task scheduling, VFI partitioning, tile mapping, routing, and voltage/frequency assignment. The task scheduling step determines the execution order of tasks assigned to the same PE. Since the task scheduling step converts the task graph G to G' by augmenting additional edges, we describe it with the function $O : G \to G$. The VFI partitioning determines which PEs have the same voltage and frequency and can be described with the function $\Phi : R \to \Pi$.

Each PE is then assigned to one of tiles in the NoC. The function $\Psi : R \to T$ is used to represent this tile mapping step. The mapping affects the total communication load because the distances between tiles are changed depending on the mapping. The routing path between tiles is then allocated. The function $\Omega : E \to P$ is used to denote this routing path allocation step, where P is the set of link sequences. In this paper, we consider only deterministic path routing algorithms. After the routing path allocation, we set $W(\ell_i)$ to be $\sum_{\forall e_j, \ell_i \in \Omega(e_j)} w(e_j)$ for all ℓ_i. If $W(\ell_i) = 0$, the link is removed from the NoC. Under VFI architectures, we can consider two types of communication links: intra-VFI links and inter-VFI links. (For descriptive purposes, we will denote intra-VFI links and inter-VFI links by just intra-link and inter-link, respectively.) Since inter-links require mcFIFOs, they consume more energy when transferring data between different VFIs. In addition, mcFIFO increases the hardware costs.

The voltage/frequency assignment step chooses appropriate operating voltage and frequency of each VFI in order to minimize energy consumption by utilizing what would otherwise be slack time. We use the function $\Theta : \Pi \to F$ to denote the voltage/frequency assignment step, where F is the set of possible voltages/frequencies for the VFIs. We denote the voltage and clock frequency of VFI I_k as $F_v(I_k)$ and $F_f(I_k)$, respectively. The clock frequency $F_f(I_k)$ can be represented as

$$F_f(I_k) = K\frac{(F_v(I_k) - F_{vt}(I_k))^\alpha}{F_v(I_k)} \tag{1}$$

where α is a technology parameter and K is a design-specific constant [7]. $F_{vt}(I_k)$ is the threshold voltage of VFI I_k.

The problem is to find the design functions which can minimize the total energy consumption while satisfying the deadline constraint of the task graph. Therefore, we can represent the VFI-based NoC design problem as follows:

VFI-based NoC Design Problem

Given $G = \langle V, E\rangle$ and $N = \langle T, \Pi\rangle$,

find the functions O, Φ, Ψ, Ω, and Θ such that

$$En = \sum_{\tau_i \in V} En(\tau_i) + \sum_{e_{i,j} \in E} En(e_{i,j}) \text{ is minimized}$$

subject to $\forall \tau_j \in V,\ \ \rho(\tau_j) \le \mathit{deadline}.$

where $En(\tau_i)$ and $En(e_{i,j})$ are the energy consumption of task τ_i and the energy consumption of the communication between τ_i and τ_j. They can be represented as

$$En(\tau_i) = w(\tau_i) \cdot C(r(\tau_i)) \cdot F_v(\Phi(r(\tau_i)))^2 \tag{2}$$

where $C(r(\tau_i))$ is the total switched capacitances per cycle at PE $r(\tau_i)$,

$$En(e_{i,j}) = w(e_{i,j}) \cdot E_{bit}(t_i, t_j) + L_{i,j} \cdot E_{interlink} \tag{3}$$

where $t_i = \Psi(r(\tau_i))$, $t_j = \Psi(r(\tau_j))$, $E_{bit}(t_i, t_j) = \sum_{k \in P_{i,j}} (E_S(t_k) + E_L(t_k) + E_B(t_k))$.

$E_S(t_k)$, $E_L(t_k)$ and $E_B(t_k)$ are the energy consumed by each flit within the switch, link and buffer of tile t_k, respectively. $P_{i,j}$ and $L_{i,j}$ are the set of tiles and the number of inter-VFI communications on the path from tile t_i and t_j, respectively. $E_{interlink}$ is the mcFIFO overhead cost. The VFI partitioning, tile mapping and routing path allocation determine $P_{i,j}$ and $L_{i,j}$. For example, in Fig. 1, if we select the routing path of (r1, r2, r6, r10, r11) for the communication between router r1 and router r11, the number of elements of $P_{1,11}$ is 5 and $L_{1,11}$ is 2. However, the routing path of (r1, r2, r3, r7, r11) requires only one inter-VFI communication.

$\rho(\tau_j)$ is the completion time of task τ_j, which is determined as

$$\rho(\tau_j) = \max_{e_{i,j} \in E} \{\rho(\tau_i) + comm(e_{i,j})\} + d(\tau_j) \tag{4}$$

where $comm(e_{i,j})$ is the communication latency while sending $w(e_{i,j})$ bits of data between $\Psi(r(\tau_i))$ and $\Psi(r(\tau_j))$, and $d(\tau_j)$ is the computation delay of task τ_j, which can be expressed as

$$d(\tau_j) = \frac{w(\tau_j)}{F_f(\Phi(r(\tau_j)))} \tag{5}$$

3.3 Overall design flow

The VFI partitioning technique proposed in [6] merges PEs that are adjacent in the predetermined tile mapping; however, it is more profitable to group PEs with similar operating speed demands. Therefore, we first need to know the proper voltage/frequency value of each PE, which is determined by the precedence/deadline constraints and the workload of each task. The V/F assignment which finds the proper voltage/frequency value of each PE to minimize the energy consumption satisfying the deadline constraint should precede VFI partitioning. Before the V/F assignment, we should know the exact workload of each PE including the communication cost between PEs on the NoC tile structure in order to estimate the slack time of each PE. The communication cost is determined by the tile mapping and routing. In tile mapping and routing, it is better to place a PE onto the neighboring tile of other PEs that will be included in the same VFI for the effective VFI implementation. Therefore, these three design steps (VFI partitioning, V/F assignment and tile mapping/routing) interact circularly as shown in Fig. 3(a).

In consideration of the above, we propose the design flow shown in Fig. 3(b) for VFI-based NoC systems. The design flow consists of six optimization steps. First, it determines the order of tasks mapped on each PE. Second, an initial (tentative) voltage and frequency is determined for each PE. Since we do not know the communication distance between PEs before network assignment, we use an optimistic approach which assumes that all communication distances are one hop. This approach is reasonable because the following design steps attempt to minimize the communication distance. Note that the voltage and frequency of each PE may change in the VFI partitioning and island V/F assignment steps.

Third, the PEs are partitioned into m number of VFIs based on the voltage and frequency values. The VFI partition is determined such that the total energy consumption for computation and communication be minimized. Since the inter-link cost is larger than the intra-link

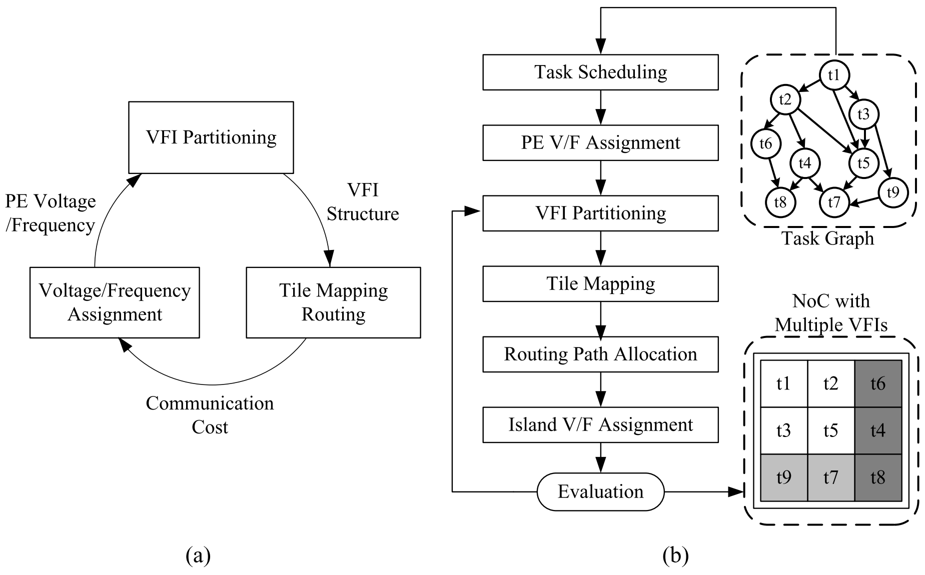


Fig. 3 Overall design flow for a VFI-based NoC

cost, it is important to reduce inter-VFI communications when determining VFI partitions. However, the previous technique proposed in [8] ignores the communication cost between VFIs since it cannot know the exact communication cost before network assignment. Our communication-aware partitioning technique, on the other hand, estimates the communication cost assuming one hop distance. This assumption is feasible since we optimize the tile mapping such that two PEs having significant communication between them are to be adjacent. In order to minimize the communication cost between VFIs, we use an iterative graph partitioning heuristic similar to the K&L algorithm [29].

Fourth, the tile mapping and routing are determined to minimize the communication cost between PEs. The VFI-aware tile mapping in [8] determines the tile location of each PE in the order of communication traffic of the PE to minimize the distance between two tiles that communicate large amounts of data. However, PEs with small communication costs can be isolated with other PEs of its VFI. To solve the problem, the technique performs the time-consuming pair-wise swapping to remove the isolated tile. In practice, it performs pair-wise swapping until it finds the optimal solution, and so, it is not an efficient heuristic as it does not prevent the exhausting search for total solution space. However, we solve the problem using the isolation-preventing tile mapping (IP-TM) technique, which reserves the space for unmapped PEs.

In routing, it is important to minimize the number of inter-links because they require mcFIFOs. The proposed algorithm uses the global inter-link allocation (GLR) technique to minimize the number of inter-links while a local inter-link allocation (LLR) technique was used in [8].

Finally, the voltage and frequency of each VFI are re-examined using the exact communication cost fixed by the tile mapping and routing steps. Therefore, the V/F assignment has two phases. If the energy of the final solution exceeds the energy constraint, we increase the number of allowable VFIs and then repeat the loop starting from *VFI Partitioning* step in Fig. 3(b). Although the exact communication delay is known after the first iteration, the sec-

Table 1 Comparison between two VFI-aware design schemes

Design step	Jang [8]	Proposed
Task model	Deadline for each task	Deadline for task graph
V/F assignment	Lowest possible V/F	Two-phase
		(PE V/F, Island V/F)
VFI partitioning	Comm-unaware	Comm-aware
	Full search	Iterative graph partitioning
Tile mapping	VFI-aware TM	VFI-aware and
		isolation-preventing TM
Routing	Local inter-link allocation	Global inter-link allocation

ond iteration should perform the VFI partitioning with the best-case communication delay since it uses a larger number of VFIs.

The overall differences between our approach and the work of Jang [8] are summarized in Table 1. In the design technique of Jang [8], each task is assigned the clock frequency which is the lowest possible value to complete its execution before its deadline since they assume each task has its deadline. However, we use a more practical task graph model as described in Sect. 3.1.

4 Detailed descriptions of VFI-based NoC design

4.1 Task scheduling and V/F assignment on PEs

For task scheduling, we adopted a list scheduling algorithm which uses the mobility of each task to determine its priority. The mobility of a task is defined as the difference between the ASAP (as-soon-as-possible) start time and the ALAP (as-late-as-possible) end time. In order to get these time values, the communication delays for respective edges are needed. However, the distance between two tiles is not available ahead of the tile mapping and routing steps. Moreover, the communication delay at a link is dependent on how many communication loads share the link. To cope with this problem, we use the best-case communication delay assuming the distance between two tiles is one hop and each communication uses its links exclusively.

After the task scheduling, the voltage and clock frequency of each PE are determined considering the deadlines of tasks and the workload of the PE. The PE V/F assignment problem is a nonlinear inequality constrained problem. Schmitz and Al-Hashimi [26] proposed a voltage and clock frequency selection algorithm similar to the gradient-descent approximation algorithm. Their algorithm first estimates the slack time of each task considering the deadline and precedence constraints. It then calculates $\Delta En(\tau_i)$ for a task τ_i which has a slack time. $\Delta En(\tau_i)$ is the energy gain when the time slot for τ_i is increased by Δt (with a lower clock speed). $\Delta En(\tau_i)$ is called an energy gradient of task τ_i. After increasing the time slot for the task τ_i with the largest $\Delta En(\tau_i)$ by a time increment Δt, it repeats the same sequence of steps until there remains no task with slack time. Since it adjusts the task with the largest power variation, it is called the *power variation* DVS algorithm. Luo and Jha [30] proposed a more efficient slack allocation technique which updates the execution times of multiple tasks at the same time when their energy gradients are at about the same level.

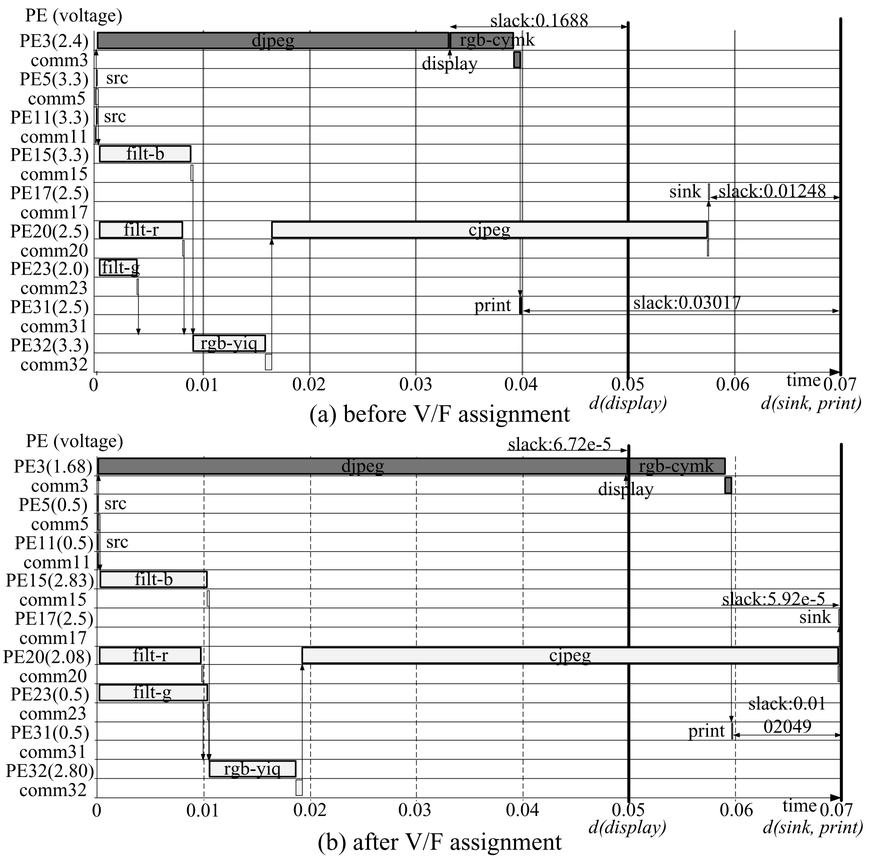


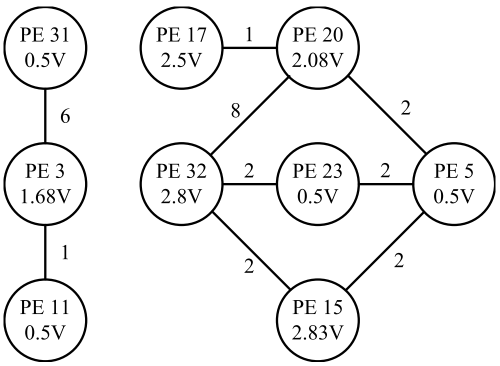
Fig. 4 Task scheduling and first V/F assignment

While the algorithms of [26, 30] determine the operating speed of each task assigned on the DVS-enabled PE, our PE V/F assignment algorithm determines an operating speed for each PE that is not changed dynamically at run time. Therefore, we calculate the energy gradient $\Delta En(r_i)$ for PE r_i which has a slack time. The slack time of r_i is the minimum among the slack times of tasks assigned to the PE r_i. For example, we can get the task schedule shown in Fig. 4 for the task graph in Fig. 2. The task schedule is driven based on the energy and timing parameters provided in E3S benchmark suites [28]. Since PE3, which should execute the tasks `djpeg` and `rgb-cymk`, has a slack time, its voltage is changed to 1.68 V. Since there are the lower limits of clock and voltage for each PE, some tasks have slack times even after the V/F assignment.

4.2 VFI partitioning

Before the VFI partitioning, we create a PE communication cost graph $G_{PE}(V, E)$ as shown in Fig. 5. Each node represents a PE with an assigned voltage and each undirected edge represents the total communication cost between two PEs. Then, we use Algorithm 1, which

Fig. 5 $G_{PE}(V, E)$ graph



Algorithm 1: VFI-Aware Partitioning

```
Input: $G_{PE}(V,E)$, m (the number of VFIs)
 1: find the lowest voltage ($L_v$) and the highest voltage ($H_v$) for all $V_{dd}(r_i)$;
 2: $\Delta V = (H_v - L_v)/m$;
 3: setup an initial VFI partition $\Pi_{cur}=\{I_1, \cdots, I_m\}$ such that $I_k$ includes $r_i$
    where $L_v \le V_{dd}(r_i)$ and ($V_{dd}(r_i) < L_v + k\Delta V$ ($i<m$) or $V_{dd}(r_i) \le L_v + k\Delta V$ ($i=m$));
 4: setup the voltage of all VFIs such that $V_{dd}(I_k) = \max\{V_{dd}(r_i) \mid r_i \in I_k\}$;
 5: get $En_{cur}$ for the current VFI partition;
 6: get the list of inter-VFI communications and sort them in descending order;
 7:   for each inter-VFI communication do
 8:     identify two source and destination PEs $r_i$ and $r_j$ of the inter-VFI comm.;
 9:     if moving $r_i$ into the VFI $\Phi(r_j)$ does not change the order of VFIs then
10:         get a modified partition $\Pi'$ by moving $r_i$ into the VFI $\Phi(r_j)$;
11:         setup the voltage of all VFIs in $\Pi'$ and get $En'$ for $\Pi'$;
12:     end if
13:     if moving $r_j$ into the VFI $\Phi(r_i)$ does not change the order of VFIs then
14:         get a modified partition $\Pi''$ by moving $r_j$ into the VFI $\Phi(r_i)$;
15:         setup the voltage of all VFIs in $\Pi''$ and get $En''$ for $\Pi''$;
16:     end if
17:     if $En_{cur} > En'$ or $En_{cur} > En''$ then
18:         set $\Pi_{cur}$ as the partition which has a lower energy between $\Pi'$ and $\Pi''$;
19:         goto line 6;
20:     end if
21:   end for
22: build the partitioned graph $G_{PE}'(V,E,\Pi_{cur})$;
Output: $G_{PE}'(V,E,\Pi)$
```

is similar to the K&L heuristic [31]. We first divide all PEs into m number of initial VFI partitions, $I_1, \ldots, I_m$. When the lowest and highest PE voltages assigned by the PE V/F assignment step are L_v and H_v, respectively, I_k includes the PEs whose voltages are greater than or equal to L_v and less than $L_v + \Delta V \cdot k$ where ΔV is $(H_v - L_v)/m$. The PE with voltage H_v is included in I_m (line 3). Since the clock frequency and voltage value of all PEs within a VFI should be identical, the values of all PEs are changed to the maximum value

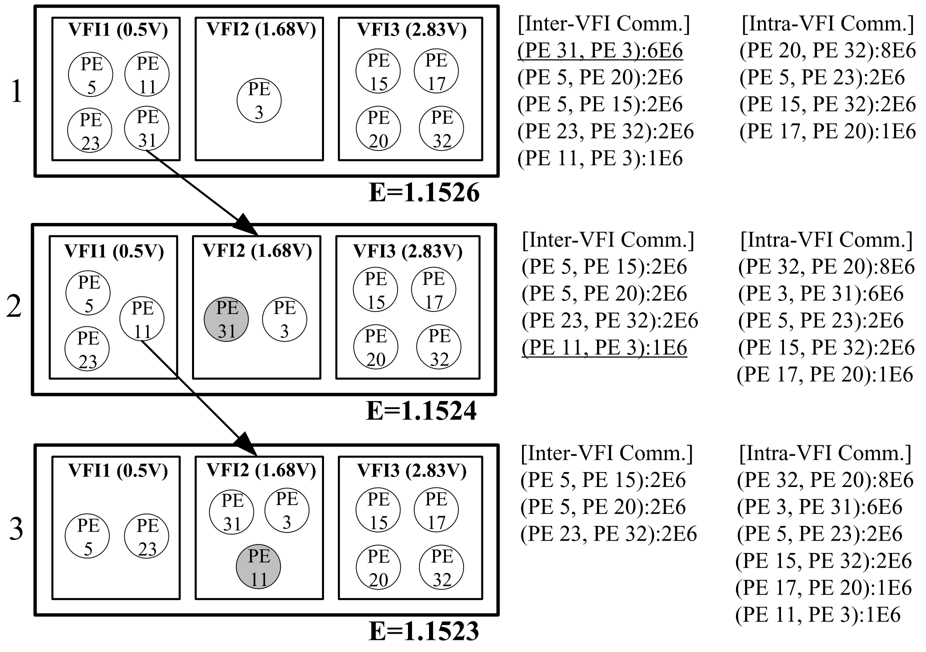


Fig. 6 Communication-aware VFI partitioning

among them, which are the voltage and clock frequency of the VFI (line 4). Figure 6 shows the initial state and changes of VFI partition assuming the number of VFIs is 3 ($m = 3$) for the example in Fig. 5.

Next, we find the communication edge with the largest inter-VFI communication cost and check whether we can reduce total energy consumption by moving one of the PEs of the communication edge so as to change the inter-VFI cost into an intra-VFI cost (line 10 and line 14). For example, since the inter-VFI communication between PE 31 and PE 3 has the largest value at the initial step partition in Fig. 6, we try to move PE 31 into VFI2 or move PE 3 into VFI1 at the first iteration. At the second iteration, the algorithm tries to remove the inter-VFI communications between (PE 5, PE 15), (PE 5, PE 20), or (PE 23, PE 32). However, none of the alternatives can reduce the total energy consumption, so the algorithm tries to remove the next highest cost. By moving PE 11 into VFI2, we can reduce energy consumption. When there remains no PEs that can reduce energy consumption by rearrangement, we stop the PE migration and get a final VFI partition solution.

When the algorithm moves a PE, it permits only those movements that do not destroy the VFI boundaries (line 9 and line 13). For example, for the third solution in Fig. 6, if PE 20 is moved into VFI1, the voltage of VFI1 should be changed into 2.08 V which is the largest voltage among those of PE 5, PE 23 and PE 20. Then, the voltage of VFI1 becomes larger than that of VFI2 and the VFI boundaries are destroyed. Therefore, we exclude such movements.

4.3 VFI-aware tile mapping

In the tile mapping step, we determine the tile location of each PE such that the PEs comprising a VFI be adjacent in the tile structure. We use Algorithm 2, which is similar to the

Algorithm 2: Isolation-Preventing and VFI-Aware Mapping

```
Input: $G_{PE}'(V,E,\Pi)$, NoC topology
 1: for all node $v_i$ do  $c(v_i) = \Sigma w(e_{i,j})$; end for
 2: sort $c(v_i)$ in decreasing order;
 3: for all candidate lists $CT(I_1), \cdots CT(I_m)$ do
 4:   $CT(I_k)$ =empty;
 5: end for
 6: for all sorted $v_i$ do
 7:     if $CT(\Phi(v_i))$ is empty then
 8:         map $v_i$ on any empty tile $t$ with max. neighbors and min. traffics;
 9:     else then
10:         map $v_i$ on any candidate tile $t$ with min. traffics in $CT(\Phi(v_i))$;
11:     end if
12:     if the mapping generates an isolated island then
13:         unmap $v_i$ and remove $t$ from $CT(\Phi(v_i))$;
14:         goto 7;
15:     end if
16:     add neighbor empty tiles of $t$ into $CT(\Phi(v_i))$;
17: end for
Output: $G_{PE}''(V,E,\Pi)$ mapped onto $N(T,m)$
```

heuristic proposed in [8] but uses a different isolation-preventing technique. PEs are sorted in decreasing order by the amount of traffic and then are mapped in this order (lines 1 and 2). There is a candidate tile list $CT(I_k)$ for each VFI I_k. The mapping algorithm selects one of the candidate tiles for the current PE being mapped by examining the candidate tile list (line 10). If no candidate exists, we select the empty tile which has the maximum number of neighbor tiles and is close to the other tiles in $CT(I_k)$ generating the minimum communication traffic (line 8). After determining the tile location of a PE, neighboring empty tiles are inserted into the candidate list (line 16). This repeats until all PEs are mapped onto the NoC grid.

While the algorithm is mapping each PE, it checks whether there is an isolated island, which consists of free tiles that cannot afford to contain all unmapped PEs of its neighbor VFIs. If an isolated island can be generated by the tile mapping of a PE, then the proposed isolation-preventing algorithm searches other candidate locations until no isolated island is generated (lines 12–15). If it cannot find an alternative location, it re-maps the lastly mapped PE. While the tile mapping algorithm in [8] uses a post-pass pair-wise tile swapping technique to remove isolated tiles, the proposed algorithm prevents generating isolated VFIs at each PE mapping and thus can achieve enhanced energy efficiency and provide a reasonable searching time.

Figure 7 illustrates the difference between isolation-unaware tile mapping and isolation-preventing tile mapping. The intermediate tile mapping after PE 10 is mapped to the tile t14 is shown in Fig. 7(a). The candidate tile list for VFI I_2 is $\{t_8, t_{11}\}$, which are neighboring empty tiles. An isolation-unaware mapping algorithm may select tile t_8 for PE 11, as shown in Fig. 7(b), and generate two separate islands for I_1, as shown in Fig. 7(c). However, the isolation-preventing mapping selects tile t_{11}, as shown in Fig. 7(d), and composes VFI I_1 with adjacent tiles as shown in Fig. 7(e). Even though the isolation-unaware mapping uses the pair-wise tile swapping process, it may deteriorate the effect of the VFI-aware tile mapping since the swapping is performed only to remove isolated islands.

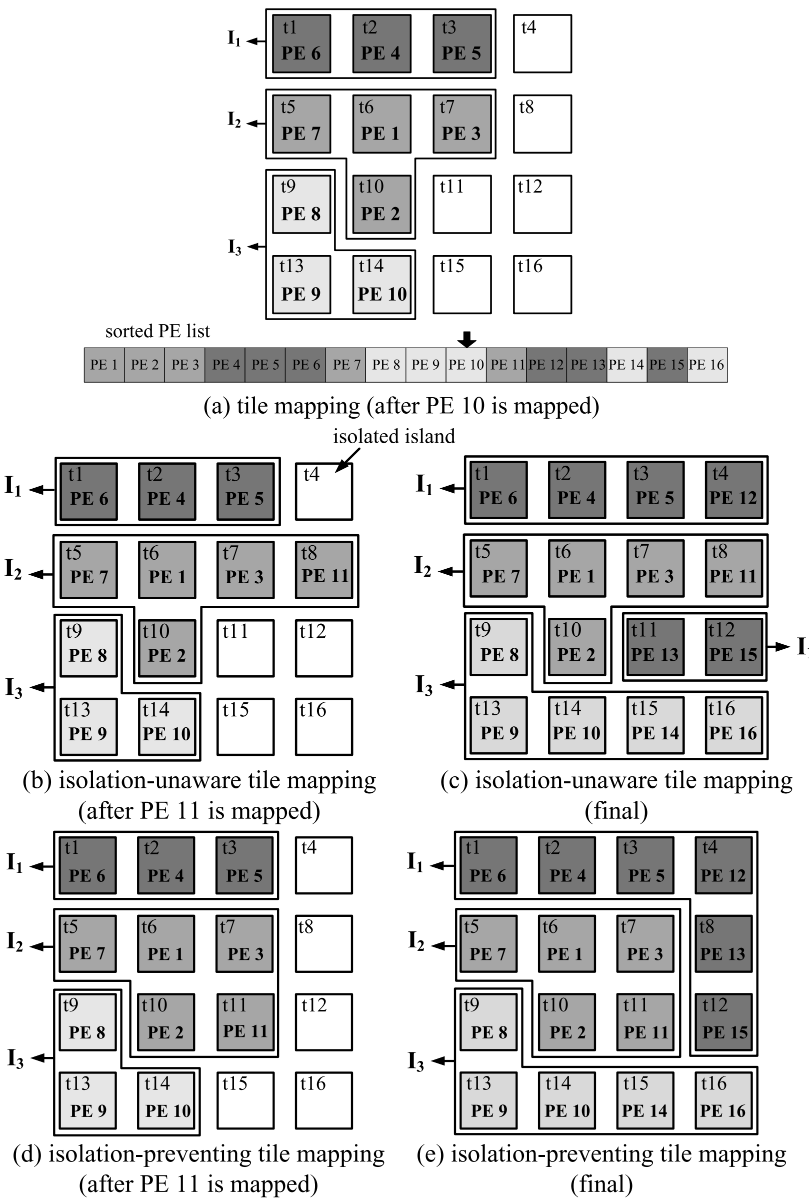

I1
I2
I3
t1 PE 6
t2 PE 4
t3 PE 5
t4
t5 PE 7
t6 PE 1
t7 PE 3
t8
t9 PE 8
t10 PE 2
t11
t12
t13 PE 9
t14 PE 10
t15
t16
sorted PE list
PE 1 PE 2 PE 3 PE 4 PE 5 PE 6 PE 7 PE 8 PE 9 PE 10 PE 11 PE 12 PE 13 PE 14 PE 15 PE 16
(a) tile mapping (after PE 10 is mapped)
isolated island
t8 PE 11
(b) isolation-unaware tile mapping (after PE 11 is mapped)
t4 PE 12
t11 PE 13
t12 PE 15
t15 PE 14
t16 PE 16
(c) isolation-unaware tile mapping (final)
t11 PE 11
(d) isolation-preventing tile mapping (after PE 11 is mapped)
t8 PE 13
(e) isolation-preventing tile mapping (final)

Fig. 7 IU-TM (isolation-unaware tile mapping) vs. IP-TM (isolation-preventing tile mapping)

4.4 VFI-aware routing

The aim of the proposed routing path allocation is to build the minimum number of inter-links as well as to minimize the total communication cost. Even though non-minimal routing paths can minimize the number of inter-links, we consider only minimal routing paths since the problem space including non-minimal routing paths is too large as well as they generally have long interconnection distances. XY-routing is a commonly used minimal routing

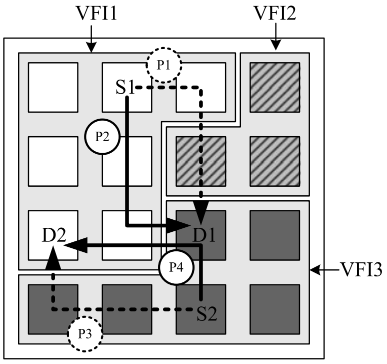


Fig. 8 Routing path allocation in a VFI-based NoC

technique in NoC design, where packets are first routed along the X-axis. Once a packet reaches the column under which the destination tile is located, it is then routed along the Y-axis. XY-routing path is one among multiple shortest paths. Due to its simplicity and good performance, XY-routing is an acceptable solution when the NoC does not utilize the VFIs.

However, in VFI-based NoC, it is necessary to consider the inter-VFI communication overhead in determining the routing path. For example, in Fig. 8, there are three VFIs and tile S1 in VFI1 should send data to tile D1 in VFI3. XY-routing uses the routing path P1, which passes through VFI2. The path P1 requires two inter-links thus has a long delay time. If we use the routing path P2 instead, only one inter-link is necessary. Therefore, the routing path should be allocated aiming to minimize the number of inter-links. Furthermore, the possibility of inter-link sharing should be taken into account. For the communication from S2 to D2, paths P3 and P4 both require one inter-link. However, P4 can share the inter-link with P2. With the help of inter-link sharing, we can reduce the hardware cost as long as the inter-link can provide the required bandwidth.

For this purpose, we use the global link allocation technique shown in Algorithm 3, which determines the required inter-links examining all communications between VFIs to globally minimize the number of inter-links. The previous routing algorithm in [8] allocates at least one inter-link at each boundary between two VFIs.

The global link allocation first composes candidate inter-links between all adjacent tiles that are located at the boundary of different VFIs (line 2). Using the PE communication graph $G''_{PE}(V, E)$ generated by tile mapping, it composes the inter-VFI communication list called *InterComm* (line 3). For each candidate inter-link (l_i), the algorithm finds the shortest path for each inter-VFI communication (e_j) that uses the inter-link (line 7). If there is such a path, it adds the required bandwidth $(w(e_j))$ to the required maximum bandwidth of the link $(W(l_i))$ and inserts the path into the set of routing paths of the inter-link $(l_i.path)$ (line 9). After estimating the maximum bandwidths and paths assigned to all candidate inter-links, the algorithm finds a candidate inter-link with the largest required bandwidth (line 13). Then, it selects the inter-link as the final one and fixes all routing paths which use the inter-link (lines 17–19). Before fixing routing paths, it adjusts the required bandwidth of the inter-link if it is larger than the maximum allowable (lines 14–16). The algorithm repeats these steps until the route paths for all inter-VFI communications are determined.

Figure 9 shows the difference between local inter-link allocation and global inter-link allocation when three inter-VFI communications are required $((S1, D1)$, $(S2, D2)$ and $(S3, D3))$. While the local approach allocates one inter-link between all different VFIs, the proposed global method allocates only two inter-links and thus enhances their utilizations as well as reduces the routing distance.

Algorithm 3: VFI-Aware Routing Path Allocation

```
Input: $G_{PE}''$(V,E), $N(T,m)$
 1: connect all tiles within each VFI;
 2: make candidate inter-links between all adjacent tiles
      which are located at the boundary of different VFIs;
 3: get the inter-VFI comm. list InterComm from $G_{PE}''$(V,E);
 4: while there is an element in InterComm do
 5:   for all candidate inter-links $l_i$ do
 6:     for all inter-VFI comm. in InterComm $e_j$ do
 7:       find a route path which is a shortest path and uses $l_i$ ;
 8:       if (found) then
 9:         add $w(e_j)$ into $W(l_k)$ and insert the path in $l_i$.path;
10:       end if
11:     end for
12:   end for
13:   find the candidate inter-link $l_k$ with the largest value of $W(l_k)$;
14:   while ($W(l_k) > BW_{max}$) do
15:     remove some paths from $l_k$.path; // paths with many alternative paths
16:   end while
17:   remove $l_k$ from the candidate inter-links and fix it;
18:   fix all the paths in $l_k$.path;
19:   remove the corresponding inter-VFI comm. from InterComm;
20: end while
```

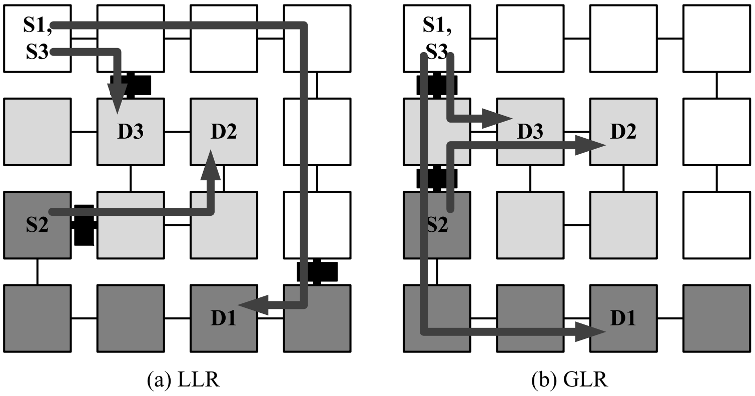


Fig. 9 Comparison between LLR (local inter-link allocation routing) and GLR (global inter-link allocation routing)

4.5 V/F assignment on VFI

Due to VFI partitioning, each PE is assigned voltage/frequency values which are larger than the values at the first-phase V/F assignment step, and thus a slack time will be generated. In addition, the network assignment determines an exact cost of each communication, which is larger than the value assumed at the first-phase V/F assignment step. As a result, several tasks may miss their deadline with the voltages and clock frequencies determined by the first-phase V/F assignment step. To resolve these problems, we should adjust the final voltage and clock frequency of each VFI at the second-phase V/F assignment step. While the

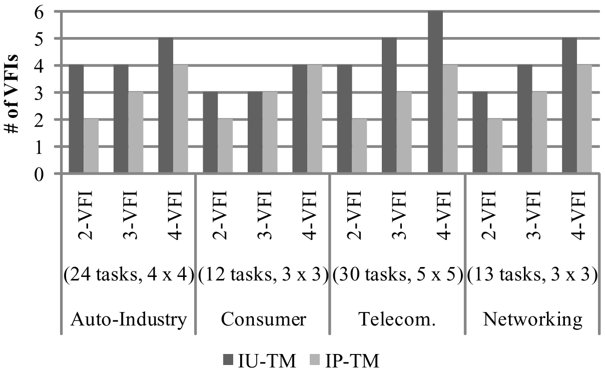


Fig. 10 Comparison of the number of VFIs

(first-phase) PE V/F assignment determines the operating speed of each PE, the (second-phase) island V/F assignment determines an operating speed for each VFI. We calculate the energy gain (or loss) $\Delta En(I_i)$ for VFI I_i which has a slack time (or missed time), that is the minimum among the slack times of tasks assigned to it. Considering the power variation of the VFI, we adjust the voltage/frequency values using the same algorithm used in the PE V/F assignment step.

5 Experiments

We estimated the effects of the proposed technique using four applications in E3S benchmark suites [28], auto-industry, consumer, telecommunication, and networking, which contains 24, 12, 30 and 13 tasks mapped onto 4×4, 3×5, 5×5 and 3×3 NoC grids, respectively. The energy and timing parameters of a combination of PE and task are derived from the benchmarks. The parameters of communications are taken from [24].

We first compared the performances of the two kinds of tile mapping techniques, IU-TM and IP-TM, explained in Sect. 4.3. Figure 10 shows the number of VFIs generated by each tile mapping technique. The X-axis is the input VFI number and the Y-axis represents the generated VFI number (counting isolated VFIs separately). Since IU-TM may generate isolated VFIs (though IU-TM uses a swapping technique to remove isolated VFI, we did not use the technique in this experiment since it takes a significant amount of time), the number of generated VFIs is larger than the input VFI number in most configurations. However, IP-TM generates the same number of VFIs as the input. The superiority of IP-TM is more significant when the number of tasks is large (i.e., auto-industry and telecommunication).

Figure 11 shows the effect of the global inter-link allocation technique explained in Sect. 4.4. The Y-axis represents the number of mcFIFOs generated by each routing technique. We assumed that the maximum bandwidth of inter-link is 10 Mbit/s. The GLR technique reduces the numbers of mcFIFOs compared to the LLR technique and increases the utilizations of inter-links. The number of mcFIFOs generated by GLR depends on the maximum allowable bandwidth of inter-link. As we increase the bandwidth constraint, the number of mcFIFOs decreases by the GLR technique as shown in Fig. 12. However, there is little change in the LLR technique since it allocates at least one inter-link at each VFI boundary.

We also compared our communication-aware design scheme with the previous research [8]. As shown in Table 2, we compared six kinds of design schemes. For the task scheduling and V/F Assignment steps, all schemes used the mobility-driven list scheduling

Fig. 11 Comparison of the required numbers of mcFIFOs between IU-TM & LLR and IP-TM & GLR

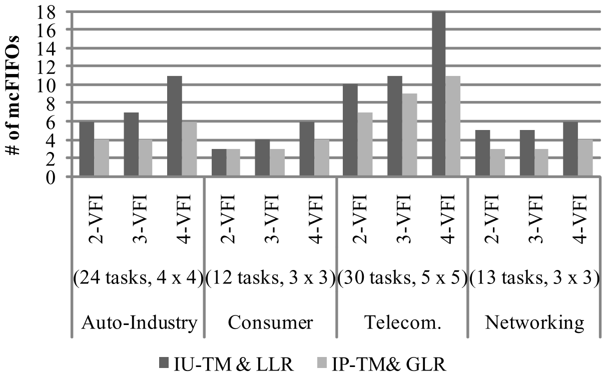


Fig. 12 Changes of the number of mcFIFOs while varying the maximum bandwidth of inter-link (Telecom. benchmark in 3 VFIs)

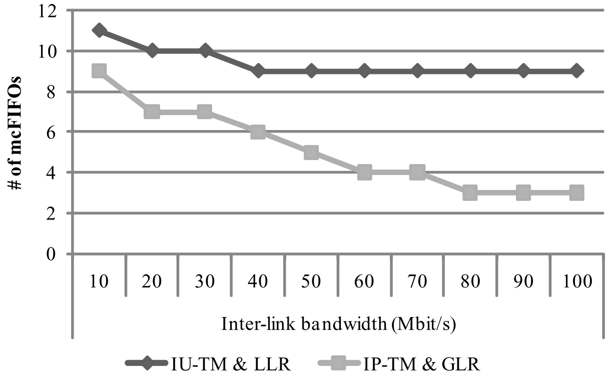


Table 2 Design schemes for experiments

Experiment	Partitioning	Mapping	Routing
Scheme1 (JJJ)	Jang [8]	Jang [8]	Jang [8]
Scheme2 (OJJ)	Proposed	Jang [8]	Jang [8]
Scheme3 (JOJ)	Jang [8]	Proposed	Jang [8]
Scheme4 (JJO)	Jang [8]	Jang [8]	Proposed
Scheme5 (JOO)	Jang [8]	Proposed	Proposed
Scheme6 (OOO)	Proposed	Proposed	Proposed

and power variation-driven V/F assignment techniques, respectively. For VFI partitioning, tile mapping, and routing steps, each scheme used either the previous technique or the proposed technique.

Figure 13 shows the energy consumptions comparison of the six design schemes for a different number of VFI configurations. The result is normalized by the energy consumption of scheme1 (JJJ version) and is divided into communication energy and PE energy.

Scheme6 (OOO version) provides a total energy consumption reduction of 3–14%. As the proposed techniques focus on reducing the communication energy, Scheme6 reduces it by 32–51%. The consumer and networking benchmarks whose communication costs are large show corresponding great reductions in total energy consumption. From the results, it is ob-

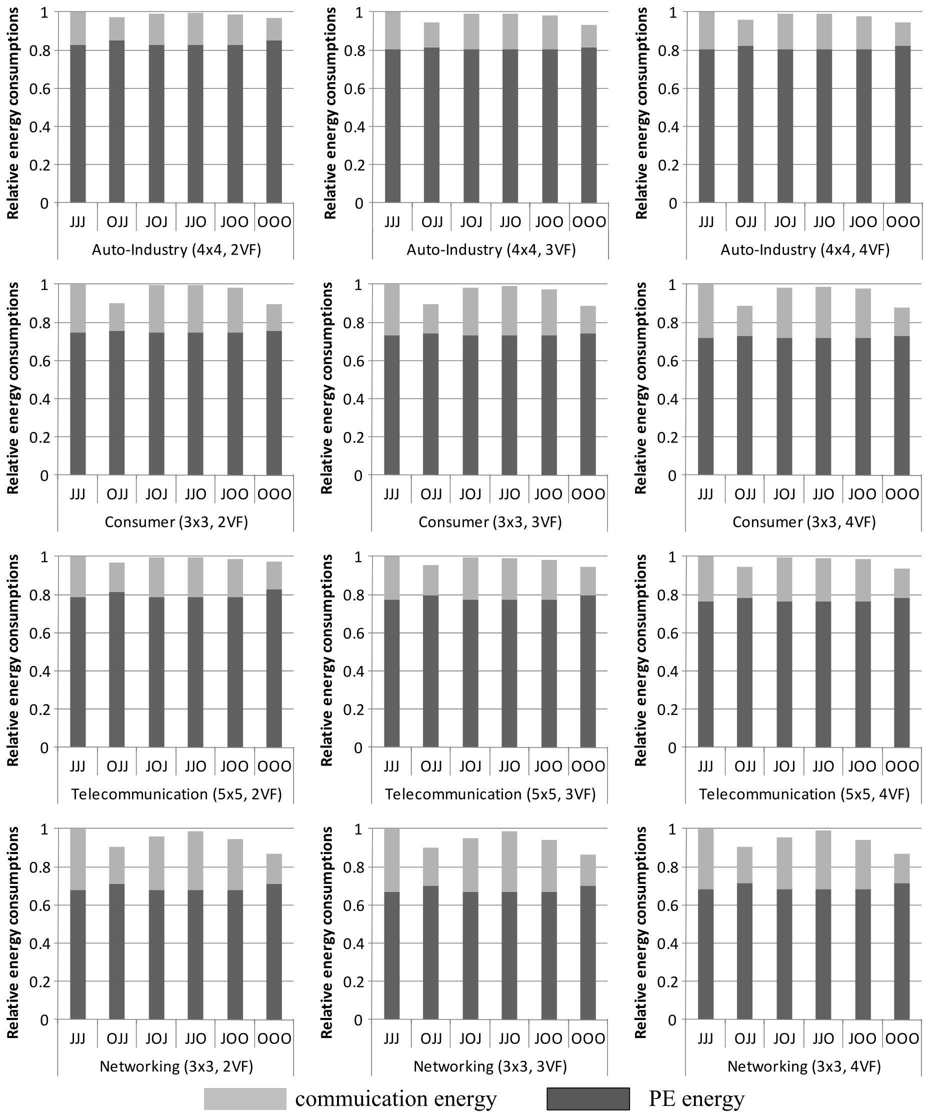


Fig. 13 Energy comparison with the previous scheme [8]

served that the main contributor on communication energy reduction is the communication-aware VFI partitioning. The IP-TM and GLR techniques contribute on reducing the number of VFIs and mcFIFOs, respectively. The effect of communication-aware partitioning grows more significantly as the number of VFIs increases.

6 Conclusions

In this paper, we proposed an energy-efficient design technique to optimize communication energy as well as computation energy in VFI-based NoC systems. The proposed algorithms optimize the energy at various design steps including VFI partitioning, tile mapping, routing path allocation, and V/F assignment. We developed a communication-aware VFI partitioning, isolation-preventing tile mapping, and global inter-VFI link allocation techniques to improve upon the limitations of previous design schemes. As a result, our algorithm reduced the communication energy consumption of VFI-based NoC systems by up to 51% compared with previous design schemes.

Our work can be extended in several directions. In this paper, we assumed that the voltage and clock frequency of VFIs do not change dynamically at run time. However, it would be more energy-efficient to provide a dynamic adaptation scheme that exploits the run-time slack. To adjust the voltages and clock frequencies of all PEs of one VFI synchronously, there should be a strong correlation among workload changes assigned to the PEs of the VFI. Considering such correlation in VFI partitioning is our future work.

Acknowledgements This research was partially supported by KORUS Tech Program funded by the KIAT (KORUSTECH(KT)-2008-DC-AP-FS0-0003).

References

1. Benini L, Micheli GD (2002) Networks on chip: a new SoC paradigm. IEEE Comput 35(1):70–78
2. Dally WJ, Towles B (2001) Route packets, not wires: on-chip interconnection networks. In: Proc design automation conference, pp 684–689
3. Chapiro DM (1984) Globally asynchronous locally synchronous systems. PhD dissertation, Dept Comput Sci, Stanford Univ, Stanford, CA
4. Hemani A, Meincke T, Kumar S, Postula A, Olsson T, Nilsson P, Oberg J, Ellervee P, Lundqvist D (1999) Lowering power consumption in clock by using globally asynchronous locally synchronous design style. In: DAC '99: proceedings of the 36th annual ACM/IEEE design automation conference, pp 873–878
5. Chelcea T, Nowick SM (2000) A low-latency fifo for mixed-clock systems. In: Proc of the IEEE computer society annual workshop on VLSI (WVLSI'00), p 119
6. Ogras UY, Marculescu R, Choudhary P, Marculescu D (2007) Voltage-frequency island partitioning for GALS-based networks-on-chip. In: DAC '07: proceedings of the 44th annual design automation conference, pp 110–115
7. Ogras UY, Marculescu R, Marculescu D, Jung EG (2009) Design and management of voltage-frequency island partitioned networks-on-chip. IEEE Trans Very Large Scale Integr Syst 17(3):330–341
8. Jang W, Ding D, Pan DZ (2008) A voltage-frequency island aware energy optimization framework for networks-on-chip. In: ICCAD '08: proceedings of the 2008 IEEE/ACM international conference on computer-aided design, pp 264–269
9. Intel, Single-chip cloud computer. http://techresearch.intel.com/articles/Tera-Scale/1826.htm
10. Shang L, Peh L-S, Jha NK (2003) Dynamic voltage scaling with links for power optimization of interconnection networks. In: Proc international symposium on high-performance computer architecture
11. Worm F, Ienne P, Thiran P, Micheli GD (2002) An adaptive low power transmission scheme for on-chip networks. In: Proc international system synthesis symposium, pp 92–100
12. Kim EJ, Yum KH, Link GM, Vijaykrishnan N, Kandemir M, Irwin MJ, Yousif M, Das CR (2003) Energy optimization techniques in cluster interconnects. In: Proceedings of the 2003 international symposium on low power electronics and design, pp 459–464
13. Soteriou V, Peh L-S (2003) Dynamic power management for power optimization of interconnection networks using on/off links. In: Proc symposium on high performance interconnects, pp 15–20
14. Hu J, Marculescu R (2003) Exploiting the routing flexibility for energy/performance aware mapping of regular NoC architectures. In: Proc design, automation and test in Europe conference, pp 10688–10693
15. Shin D, Kim J (2004) Power-aware communication optimization for networks-on-chips with voltage scalable links. In: CODES '04: proc of international conference on hardware/software codesign and system synthesis, pp 170–175
16. Marcon C, Calazans N, Moraes F, Susin A, Reis I, Hessel F (2005) Exploring NoC mapping strategies: an energy and timing aware technique. In: DATE '05: proc of the conference on design, automation and test in Europe, pp 502–507
17. Chen G, Li F, Kandemir M (2006) Compiler-directed channel allocation for saving power in on-chip networks. In: POPL '06: conference record of the 33rd ACM SIGPLAN-SIGACT symposium on principles of programming languages, pp 194–205

18. Li F, Chen G, Kandemir M (2005) Compiler-directed voltage scaling on communication links for reducing power consumption. In: Proceedings of the 2005 IEEE/ACM international conference on computer-aided design, pp 456–460
19. Kandemir M, Ozturk O (2008) Software-directed combined cpu/link voltage scaling fornoc-based cmps. In: Proceedings of the 2008 ACM SIGMETRICS international conference on measurement and modeling of computer systems, pp 359–370
20. Sheibanyrad A, Panades IM, Greiner A (2007) Systematic comparison between the asynchronous and the multi-synchronous implementations of a network on chip architecture. In: DATE ’07: proceedings of the conference on design, automation and test in Europe, pp 1090–1095
21. Bjerregaard T, Sparso J (2005) A router architecture for connection-oriented service guarantees in the mango clockless network-on-chip. In: DATE ’05: proceedings of the conference on design, automation and test in Europe, pp 1226–1231
22. Beigne E, Clermidy F, Vivet P, Clouard A, Renaudin M (2005) An asynchronous NoC architecture providing low latency service and its multi-level design framework. In: ASYNC ’05: proceedings of the 11th IEEE international symposium on asynchronous circuits and systems, pp 54–63
23. Leung L-F, Tsui C-Y (2007) Energy-aware synthesis of networks-on-chip implemented with voltage islands. In: DAC ’07: proceedings of the 44th annual design automation conference, pp 128–131
24. Guang L, Nigussie E, Koskinen L, Tenhunen H (2009) Autonomous DVFS on supply islands for energy-constrained NoC communication. In: ARCS ’09: proceedings of the 22nd international conference on architecture of computing systems, pp 183–194
25. Ghosh P, Sen A, Hall A (2009) Energy efficient application mapping to NoC processing elements operating at multiple voltage levels. In: NOCS ’09: proceedings of the 2009 3rd ACM/IEEE international symposium on networks-on-chip, pp 80–85
26. Schmitz MT, Al-Hashimi BM (2001) Considering power variations of DVS processing elements for energy minimisation in distributed systems. In: Proc international symposium on system synthesis, pp 250–255
27. Lim S, Bae Y, Jang G, Rhee B, Min S, Park C, Shin H, Park K, Kim C (1995) An accurate worst case timing analysis for RISC processors. In: IEEE transactions on software engineering, vol 21, pp 593–604
28. Dick RP (2003) Embedded system synthesis benchmarks suites. http://ziyang.eecs.umich.edu/dickrp/e3s/
29. Kernighan B, Lin S (1970) An efficient heuristic procedure for partitioning graphs. Bell Syst Tech J 49:291–307
30. Luo J, Jha NK (2007) Power-efficient scheduling for heterogeneous distributed real-time embedded systems. IEEE Trans Comput-Aided Des Integr Circuits Syst 26(6):1161–1170
31. Vahid F, Le TD (1997) Extending the Kernighan-Lin heuristic for hardware and software functional partitioning. Des Autom Embed Syst 2:237–261