Maximizing the Performance of NoC-based MPSoCs under Total Power and Power Density Constraints

Alireza Shafaei[1], Yanzhi Wang[2], Lizhong Chen[3], Shuang Chen[1], and Massoud Pedram[1]
[1]Department of Electrical Engineering, University of Southern California, Los Angeles, CA
[2]Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY
[3]School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR
shafaeib@usc.edu, ywang393@syr.edu, chenliz@oregonstate.edu, shuangc@usc.edu, pedram@usc.edu

***Abstract*—This paper presents an application mapping problem, which aims to maximize the performance of NoC-based multi-processor system-on-chip (MPSoC) designs without violating the total power and power density budgets of the chip, while maintaining the routability of all communicating cores. The mapping problem also accounts for the fact that, due to process variations, speed and leakage power characteristics of cores and routers may be quite different from one another. The problem is formulated as a mixed-integer, non-linear mathematical program, and solved heuristically by a polynomial-time combinatorial algorithm. The proposed algorithm achieves 34% (31%) on average and 52% (49%) maximum performance improvement under 16nm planar CMOS (7nm FinFET) technology when mapping different applications with different number of tasks to a 64-core processor compared with the baseline algorithms.**

I. Introduction

The multicore era has emerged as a response to the increasing demand for faster digital computation under a fixed power consumption budget. Traditionally, performance improvement was obtained by increasing the clock frequency of a single-core processor, a trend which is difficult to follow due to peak power density issues. Alternatively, in a multicore processor, the performance is enhanced by running more tasks in parallel while the frequency is kept unchanged. However, as the number of cores increases so does the total power consumption of the chip, which eventually causes a portion of the chip, known as the *dark silicon*, to be powered off [1]–[3]. In other words, only a subset of cores can be active at any time, a consequence of which is limited performance.

On the other hand, because of process variations, which are inevitable during the manufacturing process, cores may exhibit different characteristics in terms of ON/OFF currents, etc. Generally, the effect of process variations is more significant on the threshold voltage value, V_{th}, of the underlying transistors, either in bulk CMOS process, because of random dopant fluctuations, or in FinFET technology, due to the variations on the work function and line-edge roughness. The value of V_{th} almost linearly affects the ON current, as α in the alpha power-law model is approximately 1.3 for new technology nodes [4], but it exponentially impacts the OFF (leakage) current. Accordingly, speeds and more distinctly leakage power consumptions of cores may be quite different from one another. As a result, some cores, because of consuming extremely high leakage power, may not be used due to the limited power budget in the dark silicon era [1], [3].

Therefore, the dark silicon phenomenon is limiting the number of active cores in the platform, whereas core-to-core variations are limiting the application latency as well as the subset of active cores. Unfortunately, these effects become worse under sub-20nm technologies. More specifically, by scaling-down to new technology nodes and shrinking the transistor sizes, more cores can be packed on a same-area chip. However, because of the high cost of power delivery and cooling, the *thermal design power* (TDP) of processors is kept constant or even decreases with scaling to smaller device feature sizes (because per-core power density increases) [5], resulting in more portion of dark silicon. On the other hand, (i) extremely small geometries for sub-20nm technology nodes result in significant change in device properties even with slight manufacturing deviations, and (ii) reduced supply voltage, V_{dd}, levels which narrow the gap between V_{dd} and V_{th}, will exacerbate process variations and hence increase mismatches in core-level characteristics. Consequently, under deeply-scaled technologies, ignoring the impact of dark silicon and core-to-core variations may result in significant performance degradations.

Another issue associated with multicore architectures is the communication requirements among multiple tasks of an application, which are handled by a *network-on-chip* (NoC). This communication overhead, especially in processors with a large number of cores, further limits the desired performance, making the NoC a new performance bottleneck. Furthermore, the NoC consumes a substantial portion of the total chip power consumption [6]. Additionally, some routers, even though their corresponding core is powered off, must remain active in order to maintain the routability, which in turn increases the ratio of the NoC power to the total chip power consumption. Therefore, routers consume a sizeable percentage of the maximum power budget of the chip [6]. On the other hand, routers are also subject to process variations, and hence, they may also experience quite different speed and leakage power behaviors from one another. In summary, power consumption and variability of routers in the NoC structure need to be considered jointly with cores when the total chip power consumption is limited in the dark silicon era.

This paper thus presents an application mapping algorithm considering the effect of both core-to-core and router-to-router variations, while constraints enforced by the dark silicon era and routability are met. More precisely, we map an application with multiple tasks to an NoC-based *multiprocessor system-on-chip* (MPSoC) such that the performance is maximized without violating the total power and power density budgets of the chip, and maintaining the routability of communicating cores. The problem is formulated as a mixed-integer, non-linear mathematical programming. The problem should be solved during runtime per application, and hence a polynomial-time heuristic is proposed to solve the optimization problem. The solution includes (i) the selection of active cores and routers, (ii) the optimal V_{dd} values of cores and routers, (iii) the optimal frequency level of routers, and (iv) the assignment of tasks to cores.

In summary, this paper presents the following contributions:

- We formally describe an application mapping problem to NoC-based MPSoCs which simultaneously considers performance maximization, power overheads, and variabilities of both cores and the NoC structure under a fixed TDP value. The problem, which is NP hard, is formulated as a mixed-integer, non-linear mathematical program.
- We present a polynomial-time heuristic to solve the aforesaid problem. Our *variation-aware, power-constrained, maximum-performance application mapping* (VPM2) algorithm optimizes the performance while constraints imposed by the dark silicon phenomenon and routability of communicating cores are not violated.
- We adopt advanced technology nodes, 16nm bulk CMOS and 7nm FinFET, to evaluate the effectiveness of our approach.

The rest of the paper is organized as follows. Section II reviews the related work. The application mapping problem is formulated in Section III. The proposed algorithm is discussed in Section IV, followed by simulation results in Section V. Finally, the paper is concluded in Section VI.

II. Related Work

As the first work to present the dark silicon phenomenon, Esmaeilzadeh et al. [1] studied the multicore scaling limits accounting for technology scaling and degree of parallelization, and according to their results, more than half of the chip must be powered off under deeply-scaled technologies. The dark silicon phenomenon is leveraged in GreenDroid [7] by filling specialized cores for common applications, and in darkNoC [8] by integrating multiple layers of routers with different speed and leakage behaviors, both in order to enhance chip's energy efficiency under certain total power budget. Thermal management and reliability issues of multicore platforms in the context of dark silicon have been addressed in [3], [9], [10].

The closest research to our work are Cherry-Picking [11] and NoC-Sprinting [6]. Process variations in a homogeneous chip multi-processor have been leveraged in Cherry-Picking [11] in order to select a subset of cores that minimizes the execution time of a multi-threaded application within a fixed power budget. However, the communication overhead of the NoC structure has been neglected. The communication overhead has been considered in the NoC-Sprinting [6], which proposes a power-efficient NoC mapping policy in the dark silicon era, but it does not consider the variations among cores or routers and only focuses on the special scenario of NoC sprinting. Our work, however, considers variability, latency, and power dissipation of the NoC structure along with those of the cores in order to provide a mapping solution with improved performance under power and connectivity constraints.

Application mapping to NoC-based MPSoCs has also been the topic of various research studies. A branch-and-bound-based mapping algorithm is proposed in [12] to minimize the communication energy under given performance constraints. NMAP [13] minimizes the average communication delay under bandwidth constraints. A distributed mapping algorithm based on agents is presented in [14]. In [15], two heuristics are proposed for mapping to MPSoCs with heterogeneous cores on irregular mesh or custom architecture. Reference [16] employs discrete particle swarm optimization for application mapping. A mapping algorithm for average packet latency minimization in express channel-based NoCs is proposed in [17]. In the aforesaid work, characteristics of the underlying routers are assumed to be identical. Moreover, recent studies have shown that power-gating idle routers, if done with careful attention to avoid frequent wakeup overheads, is an effective solution to minimizing the leakage power consumption of the NoC structure [18]–[21].

III. Problem Statement

A. System Models

Multicore Platform and Process Variations: We consider an MPSoC with N tiles, where each tile is comprised of a core and a router, and routers are arranged in a two-dimensional (2D) mesh topology. We also consider the effect of process variations, which causes speeds and leakage power consumptions of cores and routers to be different from one another [11]. Furthermore, we assume that the MPSoC supports *dynamic voltage and frequency scaling* (DVFS).

An MPSoC may have heterogeneous cores with even different functionalities [22]. Let $C^{c}_{sw,i}$ denote the switched capacitance of each core c_i. In practice these heterogeneous cores typically belong to only a few power domains, i.e., the number of power domains is much lower than N, because of the high implementation overhead (e.g., DC-DC converters, power rails) when more power domains are supported [23]. In the rest of paper, for the sake of simplicity but without loss of generality, we assume a single power domain for all cores [24], and let V^{c}_{dd} denote the supply voltage of all cores. Note that the proposed analysis and optimization framework is applicable to the case of multiple (but a small number of) power domains for cores. On the other hand, routers in the NoC structure are homogeneous and belong to a single power domain. Let C^r_{sw} and V^r_{dd} denote the switched capacitance and supply voltage of every router, respectively. Since the effect of process variations on capacitances is small, the switched capacitance C^r_{sw} is assumed to be fixed for every router.

In the MPSoC platform, core c_i has an operating frequency f^c_i at each supply voltage level V^c_{dd}. f^c_i values may be different for different cores [11], because of (i) the inherent structural heterogeneity among cores and (ii) the effect of process variations (this effect can only be captured post-manufacturing). This framework is general and is also applicable to the more common case when cores with homogeneous architecture and design in the MPSoC share the same frequency domain and f^c_i is determined by the slowest among all cores that are turned on at the time. On the other hand, routers in the MPSoC platform are homogeneous and share the same frequency domain [19], [24], i.e., they operate at the same frequency f^r at each supply voltage level, and f^r is determined by the slowest router among all the turned-on routers (due to process variations, we may encounter fast and slow routers on the same chip)

Consider that a task t_u ($1 \leq u \leq M \leq N$) is mapped to core c_i, the total power consumptions of core i, denoted by P^c_i, is calculated as:

$$P^c_i = L^c_i + \alpha^c_u f^c_i C^c_{sw,i}(V^c_{dd})^2, \tag{1}$$

where L^c_i denotes the leakage power consumption of c_i which is significantly impacted by process variations, and α^c_u denotes the activity factor of task t_u. On the other hand, the total power consumption of router i if it is turned on, denoted by P^r_i, is given by

$$P^r_i = L^r_i + \alpha^r f^r C^r_{sw}(V^r_{dd})^2, \tag{2}$$

where L^r_i denotes the leakage power consumption of router i which is significantly impacted by process variations, and we use α^r to denote the average activity factor over all routers, which is task-independent, due to the following two reasons: (i) the ratio of dynamic power consumption to leakage power consumption of NoC routers is low [18], and (ii) a NoC router is typically responsible for routing packages from different sources to different destinations, i.e., packet originators (sources) are not necessarily limited to the core that is directly connected to the router.

Dark Silicon: The overall power consumption of the MPSoC should not exceed a maximum power budget, denoted by $\mathcal{P}_{max}$. As a result, a portion of the chip is typically left dark, or in other words, only a subset of cores and routers can be powered on at any given time. Moreover, the total power consumption of each tile (i.e., a core plus its associated router) is constrained by the maximum power density, denoted by $\mathcal{P}_{density}$. In order to save power, unselected cores are completely power-gated. Unneeded routers could also be power-gated to save more power [6], [18]. However, a router that is connected to an active core must be powered on. On the other hand, the MPSoC must ensure that between any two non-adjacent cores with any data communication, a deadlock-free routing path exists. This means that a router, even if not connected to an active core, may be powered on to maintain the overall routability.

Application: During any decision epoch, we assume $M \leq N$ tasks of an application will be mapped and executed on the MPSoC. For a given application, w_{uv} denotes the average number of packets sent from task t_u to t_v per unit time, and $\psi(u) = i$ represents a function that maps t_u to c_i. Moreover, h_{ij} denotes the minimum hop count between cores c_i and c_j, which is measured based on the shortest routing path between c_i and c_j. Therefore, after mapping the application onto the MPSoC, the average packet latency, which is a metric that reflects the communication time, $\mathcal{T}_{com}$, is calculated by dividing the average hop count by the clock frequency of routers [17], as follows:

$$\mathcal{T}_{com} = \frac{1}{f^r} \cdot \frac{\sum_{u=1}^{M}\sum_{v=1}^{M} w_{uv} \cdot h_{\psi(u)\psi(v)}}{\sum_{u=1}^{M}\sum_{v=1}^{M} w_{uv}}. \tag{3}$$

On the other hand, the execution time of the application, $\mathcal{T}_{exe}$, can be expressed as a function of the mapping solution (i.e., which core will run which task), the behavior of each task, and f^c_i values. For simplicity, $\mathcal{T}_{exe}$ can be approximated as the average, or in the worst-case, as the minimum of $\frac{K}{f^c_i}$ values, where K is a curve-fitting parameter which can be derived from application profiling. In the first case, we have a maxsum problem whereas in the second case we deal with a maxmin optimization problem.

B. Variation-Aware Application Mapping Problem

Our optimization problem is defined as follows. **Given:** M tasks of an application in the current decision epoch, and an MPSoC with $N \geq M$ nodes, **find:** (i) the optimal set of turned on cores and routers, (ii) the optimal supply voltage levels of cores and routers, (iii) the optimal frequency level of routers[1], and (iv) a mapping of application tasks to MPSoC cores **such that** the overall delay is minimized (i.e., the performance is maximized) without violating the maximum power budget and the peak power density constraints of the chip, while maintaining the deadlock-free routability of all communicating cores. The mapping solution determines which cores and routers should be powered on, but additional active routers may be needed in order to maintain the connectivity and deadlock-free routability. Unselected routers are then power-gated.

The objective function, delay, is modeled as $a \cdot \mathcal{T}_{com} + b \cdot \mathcal{T}_{exe}$, where a and b denote the percentage of communication time and execution time, respectively, that can be overlapped on average, and $1 \leq a + b \leq 2$ ($0 \leq a \leq 1$ and $0 \leq b \leq 1$). For instance, $a + b = 2$ means no overlapping may occur which is the case when blocking communications are employed, $a = 1$, and $b = 0$, denote applications with independent parallel tasks, and for communication-intensive applications we can use $a = 0$, and $b = 1$. Values of a and b are learned during the lifetime of the MPSoC by using machine learning techniques [25].

Since routers share the same frequency domain, the minimum frequency level of powered-on routers will be picked as the

[1]Cores will run at their own frequency level for a given supply voltage level. Therefore, frequency levels of cores are not an optimization variable.

operating frequency of routers. Therefore, routers may be able to operate at a frequency level f^r implying that the number of turned-on routers whose frequency level is greater than or equal to f^r is at least M (please note that the inverse may not be true since additional routers may be activated to ensure connectivity and deadlock-free routability). Such frequency levels are called *possible* router frequencies, and are obtained as follows. For a given supply voltage level V^r_{dd}, let $\mathcal{F}^r = \{F^r_1, \cdots, F^r_N\}$ be the list of frequency levels of all N routers in the descending order, where F^r_i denotes the i^{th} element in the list. We search $\mathcal{F}^r$ starting from F^r_1 (i.e., the fastest router frequency level) until we find the first possible frequency, F^r_i. Finally, we set $f^r_{max} = F^r_i$ and $f^r_{min} = F^r_N$, which results in $f^r_{min} \leq f^r \leq f^r_{max}$. Higher values of f^r increase the speed of routers, but on the other hand, lower values of f^r cause more routers to run at the target frequency, which in turn, extends the search space of the mapping algorithm.

Optimization variables include (i) a binary variable x_{ui} which is 1 if task t_u is mapped to core c_i, and 0 otherwise, (ii) a binary variable y_i which is 1 if router r_i is powered-on, and 0 otherwise, (iii) integer variables V^r_{dd} and V^c_{dd}, which are the supply voltage levels of routers and cores, respectively, and (iv) an integer variable f^r which denotes the frequency level at which routers operate. The variation-aware application mapping problem is then formulated as a non-linear, mixed-integer program, which is described below:
minimize

$$a \cdot \frac{1}{f^r} \cdot \frac{\sum_{u=1}^{M}\sum_{v=1}^{M}\sum_{i=1}^{N}\sum_{j=1}^{N}(x_{ui} \cdot x_{vj} \cdot w_{uv} \cdot h_{ij})}{\sum_{u=1}^{M}\sum_{v=1}^{M} w_{uv}} + b \cdot \mathcal{T}_{exe} \quad (4)$$

subject to

$$\sum_{i=1}^{N}(y_i \cdot P^r_i) + \sum_{i=1}^{N}\sum_{u=1}^{M}(x_{ui} \cdot P^c_i) \leq \mathcal{P}_{max}, \quad (5)$$

$$y_i \cdot P^r_i + \sum_{u=1}^{M}(x_{ui} \cdot P^c_i) \leq \mathcal{P}_{density}, \quad \forall i, \quad (6)$$

$$y_i \cdot f^r_i + (1 - y_i) \cdot f^r_{max} \geq f^r, \quad \forall i, \quad (7)$$

$$(1 - y_i) \cdot \sum_{u=1}^{M} x_{ui} = 0, \quad \forall i, \quad (8)$$

$$\sum_{u=1}^{M} x_{ui} \leq 1, \quad \forall i, \quad (9)$$

$$\sum_{i=1}^{N} x_{ui} = 1, \quad \forall u, \quad (10)$$

$$x_{ui} \in \{0,1\}, \quad \forall u,i, \quad (11)$$

$$y_i \in \{0,1\}, \quad \forall u,i, \quad (12)$$

$$V^r_{dd}, V^c_{dd} \in V_{DD}, \quad (13)$$

$$f^r_{min} \leq f^r \leq f^r_{max}. \quad (14)$$

where, V_{DD} is the set of available supply voltage levels provided by the MPSoC. Please note that f^r and frequency levels of each core are themselves a function of V^r_{dd} and V^c_{dd}, respectively.

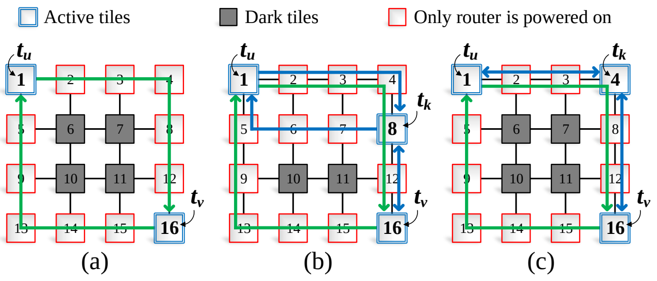

Fig. 1. Connectivity and routability constraint based on XY routing algorithm. Arrows on the figures, show routing paths between two active tiles. (a) For any two tiles with communicating tasks, additional routers on the bounding box should be powered on. However, a third task may be placed such that (b) additional routers have to be powered on, or (c) no additional router is needed.

In the optimization problem, constraint (5) ensures that the overall power consumption of the powered-on cores and routers does not exceed the maximum power budget of the chip, while constraint (6) checks the power consumption of any selected tile to be within the maximum tolerable power density. Frequency of each active router should be faster than f^r, which is addressed in constraint (7). Constraint (8) ensures that if a core is selected, the connected router should also be selected. Constraint (9) captures the fact that any core may host at most one task (some cores may not be selected at all). Moreover, each task should be assigned to only one core, as enforced by constraint (10). Constraints (11)-(14) specify the range of each optimization variable.

Deadlock-free Routability Constraint: The optimization problem also checks if any two communicating cores are interconnected through a routing path. The routing path must be selected based on a deadlock- and livelock-free routing algorithm (and not any arbitrary path), and in addition, all intermediate routers on the path must support the target frequency. As an example, without loss of generality and in order to introduce the problem, we adopt the XY routing algorithm, where packets are first sent along the X direction and then the Y direction, on a 4×4 2D mesh network, and assume that routers are running at the lowest available frequency. As shown in Figure 1(a), if tasks t_u and t_v are mapped to cores c_1 and c_{16}, respectively, and $w_{uv} \neq 0$, and $w_{vu} \neq 0$, then all routers on the bounding box of c_1 and c_{16} must be powered on. If the next task, t_k, is mapped to c_8, and has two-way communications with both t_u and t_v, then two other routers should be turned on (cf. Figure 1(b)). However, by mapping t_k to c_4, no additional router is needed, since the routing path is established through the existing powered on routers (cf. Figure 1(c)). The proposed variation-aware application mapping algorithm is applicable to other types of deadlock-free routing methods as well.

IV. Proposed Algorithm

In this section, we first discuss the challenges of our mapping problem. We then introduce the VPM2 algorithm, and analyze

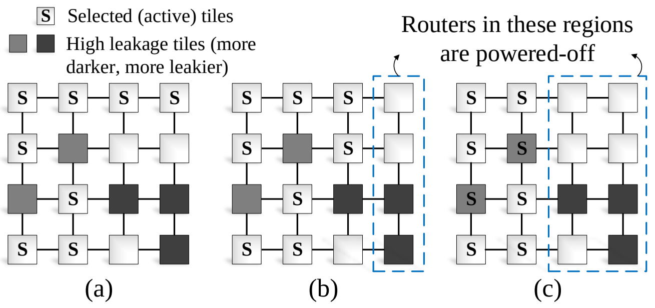
S Selected (active) tiles
High leakage tiles (more darker, more leakier)
Routers in these regions are powered-off
(a)
(b)
(c)

Fig. 2. Minimizing the number of additional routers needed to maintain XY routability. All routers in (a) must be powered on, but by changing the mapping solution (b) four, or even, (c) eight routers can be powered off. In (c), two high leakage tiles are selected, but eight routers including three very high leakage routers are powered off.

ALGORITHM 1: Variation-aware, Power-constrained, Maximum-performance application Mapping (VPM2)

```
$A_{tid}$ is the $tid^{th}$ element of list $A$. Also, $T'_1$ is the first element of
list $T'$, and its power consumption (including the power of additional
routers, if any) is denoted by $P_{T'_1}$.

procedure check_and_map() is
1      Sort $T'$ based on the total power of each node in the ascending order;
2      if ($P_{T'_1} \leq P'_{max}$) then
3          Map $A_{tid}$ to $T'_1$;   $tid++$;
4          $T' \leftarrow T' - \{T'_1\}$;   $ST' \leftarrow ST' + \{T'_1\}$;
5          $P'_{max} = P'_{max} - P_{T'_1}$;
6      else
           // Cannot find a solution
7          Exit from the for $i$ loop, and continue with the next $f$ value;

8   $\mathcal{L}_{min} = \infty$;
9   $ST = \{\}$; // Selected tiles
10  $ER = \{\}$; // Extra routers that are turned on to
        maintain the connectivity
11  $A \leftarrow$ Sort tasks based on the communication volume in the descending order;
12  foreach $v^r \in V_{DD}$ do
13      foreach $v^c \in V_{DD}$ do
14          Adjust router and core frequencies based on the
            corresponding selected supply voltage value;
15          foreach $f \in \{f^r_{min}, \cdots, f^r_{max}\}$ do
16              Temporarily set the power consumption of routers with
                   $f^r_i < f$ to $\infty$; // These routers cannot work
                   at the target frequency
17              $T \leftarrow$ List of tiles with $f^r_i \geq f$;
18              $tid$ = 1;   $P'_{max} = \mathcal{P}_{max}$;
19              $T' = T$;   $ST' = \{\}$;   $ER' = \{\}$;
20              check_and_map();
21              for $i = 2$ to $M$ do
22                  Update power of each tile in $T'$ by including the
                    routing power to each node in $ST'$. Only add the
                    power of routers that are not in $ST'$ or $ER'$;
23                  check_and_map();
24                  Remove routing powers added in line 22;
25                  Update $ER'$ list;
                // A new solution is found
26              $\mathcal{L} \leftarrow$ Find runtime of the application under the new
                configuration;
27              if ($\mathcal{L} < \mathcal{L}_{min}$) then
28                  $\mathcal{L}_{min} = \mathcal{L}$;   $ST = ST'$;   $ER = ER'$;
29                  $f^r = f$;   $V^r_{dd} = v^r$;   $V^c_{dd} = v^c$;
30                  $optimal\_mapping \leftarrow$ the new mapping solution;

31  return ($V^r_{dd}$, $V^c_{dd}$, $f^r$, $ST$, $ER$, $optimal\_mapping$)
```

its time complexity.

A. Challenges

For given V^r_{dd}, V^c_{dd}, and f^r values, the mapping problem becomes a variant of the *quadratic assignment problem* (QAP), with additional constraints enforced by the dark silicon phenomenon and the deadlock-free routability policy, which make our mapping problem more difficult than the QAP. It has been shown that QAP is NP-hard, and even no polynomial-time approximation algorithm within any factor for QAP exists, unless P=NP [26]. However, since the mapping problem is called during runtime per application, a polynomial-time heuristic is needed to solve the problem.

Same as any mapping problem, locations of the tasks must be judiciously selected. On one hand, selecting physically near tiles decreases the communication distance, which in turn reduces the communication delay. On the other hand, as indicated in Figure 2, a proper mapping solution can result in more power-gated routers. Specifically, in Figure 2(c), even though two high leakage tiles are powered on, but eight routers including three with the highest leakage power are now powered off. Therefore, fewer number of additional routers for maintaining the routability of communicating tasks is needed, resulting in reduced power consumption of the NoC. However, since cores and routers have different frequency and leakage power values, desired locations may not be available to run at the target frequency, or may have a very high leakage power consumption.

B. VPM2 Algorithm

The main idea of our proposed algorithm is to map the task with the highest communication requirement to the tile with the lowest total power consumption. In order to do this effectively, as shown in Algorithm 1, tasks are initially sorted based on their communication volume in the descending order. We then iterate on all possible values of supply voltage levels of routers and cores, as well as the frequency level of routers. For each combination, the set of tiles whose router can run at the target frequency are selected, and then, the task with the highest communication volume is mapped to the tile with the lowest total power consumption.

The same process is repeated for the remaining tasks, with the difference that the routing power (i.e. the power consumption that is needed to route data traffic between distant communicating tasks) is also considered. To this end, for each pair of remaining tiles and selected tiles, we compute the routing path based on the employed deadlock-free routing algorithm, and add the power consumption of intermediate routers to the power consumption of the respective remaining tile. However, we only add the power of routers that have not already been selected. Furthermore, for tiles whose router has
]

TABLE II
NOMINAL FREQUENCY LEVELS OF THE ADOPTED CORE, AND THE NOC ROUTER UNDER 16NM PLANAR CMOS AND 7NM FINFET TECHNOLOGIES, AND SUPER-THRESHOLD (ST) AND NEAR-THRESHOLD (NT) REGIMES. FREQUENCIES ARE REPORTED IN GHZ.

	16nm		**7nm**	
Component	**ST**	**NT**	**ST**	**NT**
Core	3	1	5	2.5
Router	2.5	0.8	3.8	2

already been turned on (to maintain the routability constraint), we only consider the core power consumption.

For routers that cannot meet the target frequency, we temporarily set their power consumption to a very large number. Accordingly, routing paths that need such routers will have a very high power consumption, and hence, are automatically excluded from the final solution. Moreover, before a mapping is committed, we make sure that the overall power consumption of the selected nodes is less than $\mathcal{P}_{max}$. Furthermore, tiles with total power consumption higher than $\mathcal{P}_{density}$ are power-gated before running Algorithm 1. Finally, the algorithm returns the best configuration that yields to the minimum runtime.

Time Complexity and Improvements: The time complexity of the $check\_and\_map()$ procedure is $O(N \cdot logN)$, due to the sort operation. However, if T' is implemented using the Min-Heap data structure, then the sort algorithm will be replaced with the `build_min_heap` operation, which is $O(N)$. On the other hand, since the MPSoC provides a fixed (and much less than N) number of V_{dd} levels, time complexities of lines 12 and 13 in Algorithm 1 are constant, $O(1)$. However, because lines 15 and 21 iterate over at most N numbers, they have a time complexity of $O(N)$. Line 22 has a time complexity of $O(N \cdot \sqrt{N})$, because the number of pairs of remaining and selected tiles is $O(N)$ (since $|T'| + |ST'| \leq N$), and for each pair, in the worst-case, the diagonal of the 2D mesh has to be checked, which is $O(\sqrt{N})$ (assuming XY routing is adopted). Therefore, the overall time complexity of Algorithm 1, VPM2, is $O(N^{3.5})$.

In the $check\_and\_map()$ procedure, we always select the remaining tile with the minimum total power consumption. However, in order to enhance the mapping solution by extending the search space, we can also try the second, third, $\cdots$, k^{th} element of list T'. If $k \ll N$, the time complexity of the algorithm will not change.

V. SIMULATION RESULTS

A. Simulation Setup

Our simulations are based on 16nm PTM planar CMOS [28] and 7nm FinFET [29] process technologies. For each technology node, we assume two supply voltage levels, namely super-threshold (ST) and near-threshold (NT) voltages. V_{dd} values for ST and NT operations in 7nm FinFET (16nm bulk CMOS) are 0.45V (0.7V) and 0.3V (0.5V), respectively. Our platform is a 64-core processor, where each tile has a 32KB L1 cache, and a 256KB, 2-bank, private L2 cache memories, both 4-way set-associative and implementing LRU replacement. An 8x8 2D mesh network is employed to connect the cores. Moreover, because of the increased power density values in deeply-scaled technologies, we adopt TDP = 30W for 16nm technology, and TDP = 15W for 7nm technology. Note that we have come up with these TDM values based on the recent results for a 36-core with 28.8W power consumption [24].

Power and Performance Parameters: Characteristics of cache memories are obtained by a modified version of CACTI tool [30], which also supports 16nm planar CMOS and 7nm FinFET devices. The NoC power is calculated using the DSENT tool [31]. The link width is 256-bit, each input port has 4 virtual channels, and each virtual channel contains five flits. We also adopted a Nehalem-based processor, whose frequency and power consumption are calculated using McPAT [32]. NoC and processor characteristics are derived under 45nm technology node, but are then scaled down to 16nm and 7nm technologies. In order to derive the appropriate technology scaling factors, we used Synopsys Design Compiler to synthesize several ISCAS bechmark circuits and processors using 45nm, 16nm, and 7nm standard cell libraries (for both ST and NT regimes).

Nominal power consumption and frequency values of the aforesaid components under 16nm and 7nm technologies are reported in Table I and Table II, respectively. As can be seen, the ST operation in both technology nodes exceeds the corresponding TDP level, which is however, more pronounced in 16nm node because of the significant increase in the short channel effects in planar CMOS devices with such small gate lengths. Based on the specified TDP values, 25 cores may be powered on in the ST regime at the same time under both 16nm and 7nm technologies. Accordingly, 60% of the chip is dark silicon which is due to the fact that the TDP level was chosen to be small. However, all cores can be simultaneously powered on in the NT operation under both technology nodes, but this comes at the cost of performance degradation due to the reduced frequency level. It is also worth mentioning that because of the improved gate control over the channel in FinFET devices, which effectively reduces the OFF current, the portion of leakage power to the total power has been reduced in 7nm FinFET compared with the 16nm planar CMOS.

Process Variation Parameters: We used the methodology described in [11] in order to model global process variations. The idea is to divide the chip into a number of grids, and to generate a correlated and normally-distributed process parameter for each grid. The process variation parameter is the gate length with a mean of 16nm (7nm) and a standard deviation of 1.6nm (0.8nm) for 16nm (7nm) process technology. Process-specific parameters to derive the delay and leakage power consumption are measured by simulating a 40-stage FO4 inverter chain in SPICE using 16nm PTM and 7nm FinFET device libraries, and curve fitting the results accordingly.

Benchmarks: We evaluate the effectiveness of our mapping algorithm by using traces from PASREC [27] benchmarks, which include blackscholes, bodytrack, canneal, dedup, flu-

TABLE I
NOMINAL POWER CONSUMPTIONS OF THE ADOPTED CORE, L1 AND L2 CACHES, AND THE NOC ROUTER UNDER 16NM PLANAR CMOS AND 7NM FINFET TECHNOLOGIES, AND SUPER-THRESHOLD (ST) AND NEAR-THRESHOLD (NT) REGIMES. P_{dyn}, P_{leak}, AND P_{tot} DENOTE THE AVERAGE DYNAMIC, LEAKAGE, AND TOTAL POWER CONSUMPTIONS, RESPECTIVELY. POWERS ARE REPORTED IN MW.

	16nm - ST			**16nm - NT**			**7nm - ST**			**7nm - NT**		
Component	P_{dyn}	P_{leak}	P_{tot}	P_{dyn}	P_{leak}	P_{tot}	P_{dyn}	P_{leak}	P_{tot}	P_{dyn}	P_{leak}	P_{tot}
Processor	274	411	685	40	234	274	151	124	275	22	70	93
L1	128	66	194	13	37	50	73	37	110	7	21	28
L2	3	129	131	1	70	72	10	93	104	1	55	56
Core	405	606	1,011	54	342	396	234	254	488	31	147	178
Router	69	118	188	10	67	78	39	67	106	6	38	44
Tile	474	724	1,198	65	409	474	273	321	595	36	185	222
64-Core			76,693			30,333			38,052			14,178

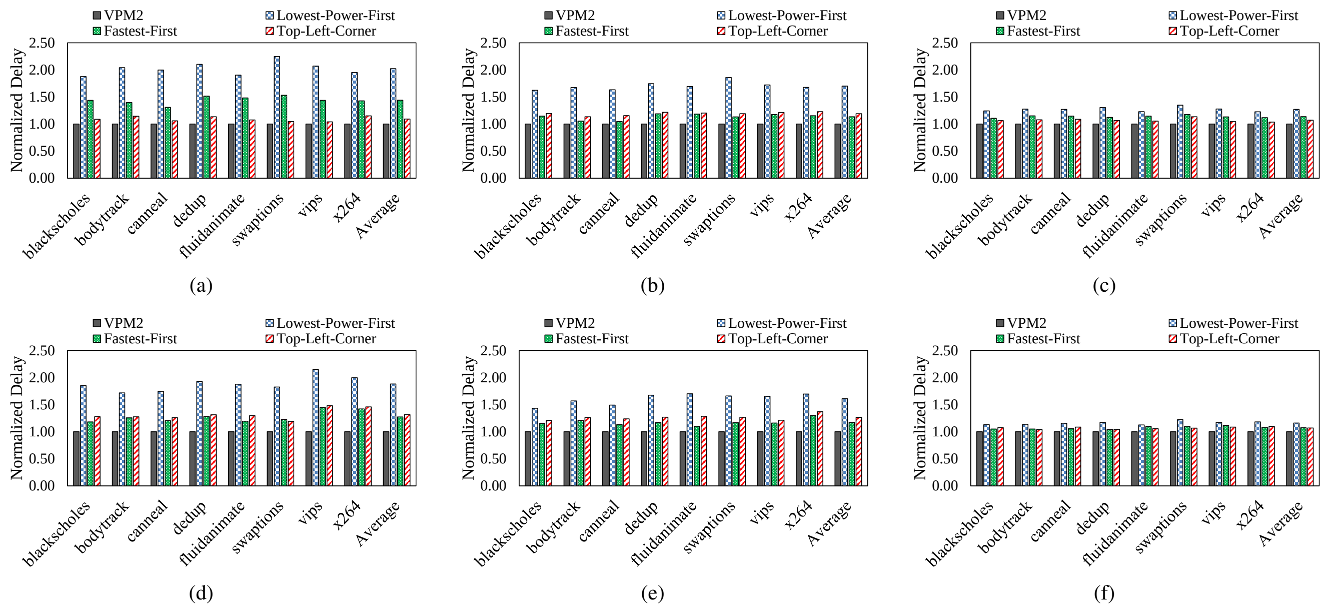

Fig. 3. Delay results for 16nm planar CMOS with (a) 24, (b) 32, and (c) 48 tasks, and 7nm FinFET with (d) 24, (e) 32, and (f) 48 tasks using different PARSEC benchmarks [27]. Results are normalized to the VPM2 results. In all cases, the Fastest-First algorithm exceeds the TDP budget, but other algorithms can find a solution within the TDP budget.

idanimate, swaptions, vips, and x264. The cycle-accurate gem5 [33] full-system simulator enhanced with GARNET [34] is used in order to generate the benchmarks.

B. Performance Improvement

We compare VPM2 with the following three baselines:

- **Lowest-Power-First:** Core with the lowest power consumption is selected first.
- **Fastest-First:** Core with the highest frequency level is selected first.
- **Top-Left-Corner:** The available tiles are limited to the smallest square on the top-left corner of the NoC that can accommodate all the tasks. Cores are selected based on joint consideration of the power consumption of the cores and the routers.

Delay results are shown in Figure 3, where the results are normalized to that of the VPM2. In all cases shown in Figure 3, the Fastest-First algorithm exceeds the TDP budget, but the results are shown for comparison purposes. Other algorithms can find a solution within the TDP budget. For 16nm planar CMOS (7nm FinFET) the average performance improvements are 52% (49%), 34% (35%), and 16% (10%) when 24, 32, and 48 tasks are mapped, respectively. If we ignore the Lowest-Power-First which does not consider the performance, then average performance improvements are changed to 27% (29%), 16% (22%), and 10% (7%) for mapping 24, 32, and 48 tasks, respectively. As we can see, in all cases our proposed algorithm outperforms the aforesaid baselines. However, by increasing the number of tasks, where fewer opportunities are available for the mapping algorithm, the performance improvement becomes smaller.

VI. CONCLUSIONS

Because of the core-to-core variabilities, and the dark silicon phenomenon, it is critical to select the optimal set of active cores in order to maximize the performance within the TDP. However, as we have shown in this paper, the locations of cores, because of the incurred performance overhead, and the additional routers needed for mainting the deadlock-free routability, which further reduce the TDP, are also important factors that should not be ignored. Accordingly, we presented VPM2 algorithm which maps tasks to an NoC-based MPSoC in order to maximize the performance without violating the TDP of the chip, and maintaining the routability of all communicating cores.

Acknowledgment

This work is supported in part by grants from the PERFECT program of the Defense Advanced Research Projects Agency, and the Software and Hardware Foundations of the National Science Foundation.

References

[1] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam, and D. Burger, "Dark Silicon and the End of Multicore Scaling," in *38th International Symposium on Computer Architecture (ISCA)*, 2011, pp. 365–376.

[2] M. B. Taylor, "Is Dark Silicon Useful?: Harnessing the Four Horsemen of the Coming Dark Silicon Apocalypse," in *49th Design Automation Conference (DAC)*, 2012, pp. 1131–1136.

[3] M. Shafique, S. Garg, J. Henkel, and D. Marculescu, "The EDA Challenges in the Dark Silicon Era: Temperature, Reliability, and Variability Perspectives," in *51st Design Automation Conference (DAC)*, 2014, pp. 185:1–185:6.

[4] T. Sakurai, "A JSSC Classic Paper: The Simple Model of CMOS Drain Current," *IEEE Solid-State Circuits Society Newsletter*, vol. 9, no. 4, pp. 4–5, Oct 2004.

[5] W. Huang, K. Rajamani, M. Stan, and K. Skadron, "Scaling with Design Constraints: Predicting the Future of Big Chips," *IEEE Micro*, vol. 31, no. 4, pp. 16–29, July 2011.

[6] J. Zhan, Y. Xie, and G. Sun, "NoC-Sprinting: Interconnect for Fine-Grained Sprinting in the Dark Silicon Era," in *51st Annual Design Automation Conference (DAC)*, 2014, pp. 160:1–160:6.

[7] N. Goulding *et al.*, "GreenDroid: A Mobile Application Processor for a Future of Dark Silicon," in *HOTCHIPS*, 2010.

[8] H. Bokhari, H. Javaid, M. Shafique, J. Henkel, and S. Parameswaran, "darkNoC: Designing Energy-Efficient Network-on-Chip with Multi-Vt Cells for Dark Silicon," in *51st Design Automation Conference (DAC)*, 2014, pp. 161:1–161:6.

[9] F. Kriebel, S. Rehman, D. Sun, M. Shafique, and J. Henkel, "ASER: Adaptive Soft Error Resilience for Reliability-Heterogeneous Processors in the Dark Silicon Era," in *51st Design Automation Conference (DAC)*, June 2014.

[10] S. Pagani *et al.*, "TSP: Thermal Safe Power: Efficient Power Budgeting for Many-core Systems in Dark Silicon," in *International Conference on Hardware/Software Codesign and System Synthesis (CODES)*, 2014, pp. 10:1–10:10.

[11] B. Raghunathan, Y. Turakhia, S. Garg, and D. Marculescu, "Cherry-Picking: Exploiting Process Variations in Dark-Silicon Homogeneous Chip Multi-Processors," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2013, pp. 39–44.

[12] J. Hu and R. Marculescu, "Energy-aware Mapping for Tile-based NoC Architectures under Performance Constraints," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2003, pp. 233–239.

[13] S. Murali and G. De Micheli, "Bandwidth-Constrained Mapping of Cores onto NoC Architectures," in *Design, Automation and Test in Europe Conference and Exhibition (DATE)*, Feb 2004, pp. 896–901 Vol.2.

[14] M. Al Faruque, R. Krist, and J. Henkel, "ADAM: Run-time Agent-based Distributed Application Mapping for On-Chip Communication," in *Design Automation Conference (DAC)*, June 2008, pp. 760–765.

[15] W. Jang and D. Pan, "A3MAP: Architecture-Aware Analytic Mapping for Networks-on-Chip," in *Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific*, Jan 2010, pp. 523–528.

[16] P. Sahu, T. Shah, K. Manna, and S. Chattopadhyay, "Application Mapping Onto Mesh-Based Network-on-Chip Using Discrete Particle Swarm Optimization," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 2, pp. 300–312, Feb 2014.

[17] D. Zhu, L. Chen, S. Yue, and M. Pedram, "Application Mapping for Express Channel-based Networks-on-Chip," in *Design, Automation and Test in Europe Conference and Exhibition (DATE)*, March 2014.

[18] L. Chen, L. Zhao, R. Wang, and T. Pinkston, "MP3: Minimizing performance penalty for power-gating of Clos network-on-chip," in *International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2014, pp. 296–307.

[19] L. Chen and T. M. Pinkston, "NoRD: Node-Router Decoupling for Effective Power-gating of On-Chip Routers," in *International Symposium on Microarchitecture (MICRO)*.

[20] R. Das, S. Narayanasamy, S. K. Satpathy, and R. G. Dreslinski, "Catnap: Energy Proportional Multiple Network-on-Chip," in *International Symposium on Computer Architecture (ISCA)*, 2013, pp. 320–331.

[21] H. Matsutani, M. Koibuchi, H. Amano, and D. Wang, "Run-time power gating of on-chip routers using look-ahead routing," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, March 2008, pp. 55–60.

[22] W. Wolf, A. Jerraya, and G. Martin, "Multiprocessor System-on-Chip (MPSoC) Technology," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 10, pp. 1701–1713, Oct 2008.

[23] Y. Choi, N. Chang, and T. Kim, "DC-DC Converter-Aware Power Management for Low-Power Embedded Systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 8, pp. 1367–1381, Aug 2007.

[24] B. Daya *et al.*, "SCORPIO: A 36-Core Research Chip Demonstrating Snoopy Coherence on a Scalable Mesh NoC with In-Network Ordering," in *International Symposium on Computer Architecture (ISCA)*, June 2014, pp. 25–36.

[25] Z. Qian *et al.*, "SVR-NoC: A performance analysis tool for Network-on-Chips using learning-based support vector regression model," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2013, pp. 354–357.

[26] S. Sahni and T. Gonzalez, "P-Complete Approximation Problems," *Journal of the ACM (JACM)*, vol. 23, no. 3, pp. 555–565, Jul. 1976.

[27] C. Bienia, "Benchmarking Modern Multiprocessors," Ph.D. dissertation, Princeton University, January 2011.

[28] W. Zhao and Y. Cao, "New Generation of Predictive Technology Model for Sub-45nm Early Design Exploration," *IEEE Transactions on Electron Devices*, vol. 53, no. 11, pp. 2816–2823, 2006.

[29] S. Chen, Y. Wang, X. Lin, Q. Xie, and M. Pedram, "Performance prediction for multiple-threshold 7nm-FinFET-based circuits operating in multiple voltage regimes using a cross-layer simulation framework," in *IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference (S3S)*, Oct. 2014.

[30] A. Shafaei, Y. Wang, X. Lin, and M. Pedram, "FinCACTI: Architectural Analysis and Modeling of Caches with Deeply-Scaled FinFET Devices," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, July 2014, pp. 290–295.

[31] C. Sun *et al.*, "DSENT - A Tool Connecting Emerging Photonics with Electronics for Opto-Electronic Networks-on-Chip Modeling," in *International Symposium on Networks on Chip (NoCS)*, May 2012, pp. 201–210.

[32] S. Li *et al.*, "McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures," in *42nd International Symposium on Microarchitecture (MICRO-42)*, Dec 2009, pp. 469–480.

[33] N. Binkert *et al.*, "The Gem5 Simulator," *SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, Aug. 2011.

[34] N. Agarwal, T. Krishna, L.-S. Peh, and N. Jha, "GARNET: A Detailed On-Chip Network Model inside a Full-System Simulator," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, April 2009, pp. 33–42.