2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems

Process Variation Aware Synthesis of Application-Specific MPSoCs to Maximize Yield

Nishit Kapadia, Sudeep Pasricha
Department of Electrical and Computer Engineering
Colorado State University, Fort Collins, CO, U.S.A.
nkapadia@colostate.edu, sudeep@colostate.edu

Abstract – In contemporary semiconductor technologies, considerable unpredictability in the behavior of manufactured chips is being observed due to the effects of process variations. This unpredictability translates into variations in power and performance within these chips. At the same time, with ever shrinking power budgets and rising cooling costs, most chip designs need to satisfy a hard limit on the maximum power that the chip can dissipate. In such a scenario, the yield of a design for a given process depends on the number of chips meeting both performance and power constraints. In this work, we propose a novel process variation-aware MPSoC synthesis framework that performs simultaneous mapping and voltage assignment of cores to mitigate the adverse effects of process variations while maximizing yield. Our experimental results show average improvements ranging from 2× to 3.8× in power-performance yield over other variation-aware MPSoC synthesis frameworks proposed in prior literature.

1. Introduction

Unpredictability in system behavior due to variability in modern fabrication processes has become a serious concern for chip designers. In modern multiprocessor system-on-chip (MPSoC) designs, spatially correlated systematic within-die (WID) variations often manifest across multiple cores, leading to core-to-core (C2C) variations [1]. At the same time, die-to-die (D2D) variations remain quite significant [2]. Both WID and D2D variations have random and systematic components. One of the primary effects of such process variations is the deviation of the threshold voltage (V_T) from its nominal value [3]. A rise in the value of V_T increases circuit delay (which decreases maximum core frequency) and decreases the leakage power of the core; on the other hand, a reduction in V_T decreases circuit delay and increases leakage power. Variations can thus impact both the performance and power of MPSoC designs in undesirable ways. Therefore there is a pressing need to mitigate process variations at all levels of chip design with new design techniques, especially at the early system-level where design decisions have the biggest impact.

In addition to unpredictability from variations, rising power dissipation is another cause for concern in today's chips. MPSoCs today almost always have an upper limit on the allowed power dissipation. Voltage islands (*VIs*), which combine cores into islands that use the same V_{DD} and ground lines, have successfully been employed to manage chip power. Also, the use of *VIs* has been shown to isolate the effects of WID process variations efficiently to boost performance [4]. One recent trend has been the rise in the power footprint of network-on-chip (NoC) fabrics. NoCs have been shown to dissipate significant power (e.g., ~30% of chip power in Intel's 80-core teraflop [5] and ~40% of chip power in MIT RAW [6] chips). As a result, reducing not only computation but also communication power has become a high priority for chip designers. In order to minimize communication power in the presence of voltage islands, efficient *VI*-aware voltage-assignment, core-mapping and routing techniques need to be employed [7]-[9], where the number of VLCs (voltage level converters) and MCFIFO (multiple clock first in-first-out) queue based frequency level converters required for inter-*VI* communication are optimized in addition to network traffic.

Given the critical need to manage variations and power in MPSoCs, new frameworks are required that can optimize chip designs for any given applications. In this paper we propose one such framework for application-specific variation- and energy-aware MPSoC synthesis. The proposed framework targets MPSoCs with a mesh-based NoC topology, similar to recent commercial MPSoCs, e.g., Tilera's TILE64 [6] and Intel's SCC [10]. Our framework improves upon prior work in several ways, with major improvements in the core voltage assignment phase. Existing work on variation-aware MPSoC synthesis (e.g., [11], [12]) typically performs core voltage assignments such that all cores meet their performance constraints for the highest (slowest) possible V_T value. But this incurs a power penalty due to higher than optimal voltage assignments to cores. To overcome the drawback of a two-step design process of first performing pessimistic voltage-assignments and then doing core-to-tile mapping as done in these works, our framework utilizes a novel variation-aware simultaneous voltage assignment and mapping technique to more efficiently reduce power while meeting performance constraints. Our experimental results show average improvements ranging from 2× to 3.8× in power-performance yield over other variation-aware MPSoC synthesis frameworks [11][12][16] proposed in prior literature.

2. Related Work

Over the past few years, the problem of variation-aware yield enhancement of MPSoCs has been studied in several works [13]-[16]. Wang et al. [13] improve performance yield by modeling task completion time as a stochastic variable and generate both task scheduling and routing procedures to optimize the probability of a given schedule meeting performance constraints. Huang et al. [15] propose to optimize performance yield by utilizing multiple test-chips representative of the spatially correlated systematic WID variations in addition to random variations. They use a simulated-annealing (SA)-based scheduling technique as well

1063-9667/14 $31.00 © 2014 IEEE
DOI 10.1109/VLSID.2014.53

as a clustering technique to generate multiple synthesis solutions at design-time, such that one of these solutions can be selected at run-time, based on the actual variation of each chip. Bhardwaj et al. [16] propose a new design metric - power-performance yield (PPY) defined as the percentage of test chips meeting both performance and power constraints. By using an SA-based approach, they trade-off the performance yield with power yield to optimize PPY. Much like prior work (e.g., [15], [16]) we consider multiple variation (V_T) maps, represented by N test chips, to capture the effects of process variations on performance and power. *But in addition to WID variations, we also consider D2D variations in our test set of N V_T-maps, which prior work [13]-[16] does not consider.*

A few prior works propose frameworks for variation-aware synthesis of MPSoCs with *VI*s, to optimize computation and communication power [11], [12]. Mazjoub et al. [11] propose a *VI*- and core-placement approach to achieve a balance between limits of spatial extent of the WID variations across cores, and communication patterns between *VI*s, by varying the shape of *VI*s to minimize total power. In our previous work [12], we consider the locations of *VI*s to alleviate the effects of process variations. Here, we make use of a single V_T-map to perform variation-aware *VI*-placement to optimize compute energy. These approaches [11], [12] tend to produce synthesis solutions with sub-optimal power because they assume pessimistic initial voltage assignments of cores corresponding to the worst-case V_T value. *In this work, for the first time, we propose a variation-aware design-time simultaneous voltage assignment and mapping based synthesis framework to reduce power while meeting performance constraints in MPSoCs with VIs. Our approach integrates novel variation-aware and VI-aware optimization techniques to produce notable improvements in PPY over prior work.*

3. Problem Formulation

The inputs to our problem are as follows:

- An MPSoC with a regular mesh-based 2D NoC with T tiles: $T = (d^2)$, where d is the mesh dimension, and each tile consists of a computate core, a NoC router, and a network interface (*NI*) between the router and core;
- A set of N V_T-maps incorporating the effects of WID and D2D variations with continuous distribution over the die;
- A core graph *G(V, E)* with a set of T vertices, representing homogeneous cores on which tasks have already been mapped and $\{f_1, f_2, f_3, \ldots, f_T\}$ representing rated operating frequencies of the T cores; and the set of M edges $\{e_1, e_2, e_3, \ldots, e_M\}$, where edge weights $\{w(e_1), w(e_2), \ldots, w(e_M)\}$ represent minimum bandwidth constraints between cores;
- A set S of candidate supply voltage (V_{dd}) levels, where each V_{dd} level can form at most one *VI*;
- A chip-wide power dissipation constraint (*PC*).

Objective: Given the above inputs, the goal of our work is to perform: *(i)* voltage assignments of cores; *(ii)* core-to-tile mapping; and *(iii)* synthesize a regular 2D mesh NoC for a specific application; such that all cores within individual *VI*s are contiguously placed while maximizing power-performance yield (PPY). We define PPY as the number of test-chips (out of N) that *(i)* satisfy frequency (performance) constraints for all T cores, as well as *(ii)* satisfy the power constraint (PC), considering power consumption (dynamic and leakage) in compute cores and communication resources (routers, links and VLCs/MCFIFOs).

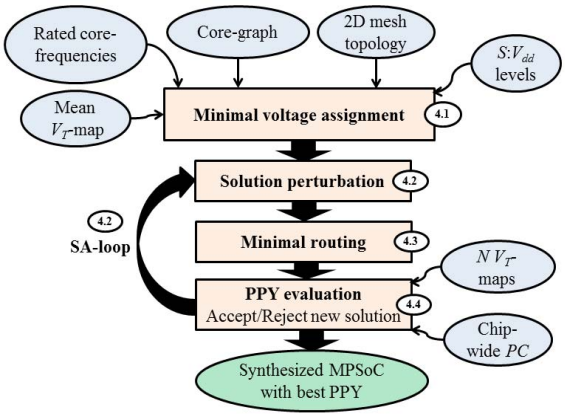


Figure 1: Design flow of the synthesis framework

4. Variation-Aware MPSoC Synthesis Framework

Figure 1 shows the flow of our proposed variation-aware MPSoC synthesis framework. Given the application core-graph and a 2D mesh of tiles on the MPSoC, we start with a core-to-tile mapping solution, which is optimized for communication traffic using incremental swapping (first proposed in [26]). Then contiguous *VI*s are formed (from the candidate V_{dd} levels of the input set S) with minimal voltage assignments for the current mapping, such that the rated frequencies of all cores are satisfied for the mean V_T-map (discussed in section 4.1). The mean V_T-map represents the test-chip (out of N test-chips) with a V_T-map which is least different from all other V_T-maps (i.e., has the least average difference in V_T values over all T tiles). A simulated annealing algorithm (SA-loop) is initialized with this synthesis solution, which iteratively perturbs the current mapping solution, performs minimal routing, and evaluates the current PPY (discussed in section 4.2). At each iteration within the SA-loop, our minimal routing heuristic allocates paths for communication flows such that the number of VLCs used for inter-*VI* data communication is minimized (discussed in section 4.3). Then, the current synthesis solution is subsequently evaluated for each of the N test-chips to obtain a corresponding PPY value. The PPY evaluation basically computes the number of test-chips which satisfy both the performance and power constraints for the given synthesis solution (discussed in section 4.4). Finally, our framework outputs the MPSoC synthesis solution with the highest PPY.

In the following sections (Sections 4.1 - 4.4), we discuss the details of each step in our synthesis framework in detail.

4.1 Minimal voltage assignment

The objective of this step is to assign minimal voltages to cores for the current core-to-tile mapping such that all cores can be clocked at their rated operating frequencies for the mean V_T-map. Note that the maximum operating frequency can be increased by increasing the supply voltage (V_{dd}), though at

the cost of increased power. Thus, for a certain $\{V_T, V_{dd}\}$ pair, the core can operate at a certain maximum frequency, in other words, to meet the minimum operating frequency requirement of a particular core, an appropriate $\{V_T, V_{dd}\}$ pair needs to be chosen. The following equation gives the relationship of core frequency f, supply voltage V_{dd}, and the V_T value of the tile, where α and β are constants.

$$f = \frac{(V_{dd}-V_T)^{\alpha}}{\beta . V_{dd}} \quad \ldots (1)$$

A valid voltage assignment must also satisfy the *VI*-contiguity constraint (all cores within individual *VI*s are contiguously placed), and voltage levels must be chosen from the input set *S*. These objectives are accomplished in two steps: *(i)* calculate minimum core voltages to satisfy performance constraints, based on the mean V_T-map (section 4.1.1); *(ii)* create up to *S* contiguous *VI*s based on the continuous distribution of minimum voltages (section 4.1.2).

4.1.1 Calculation of minimum core voltages

The minimum core voltages are calculated such that all operating frequency constraints are satisfied for V_T values of the mean V_T-map. We first elaborate on how the mean V_T-map is chosen out of the N V_T-maps. A V_T-map is represented as $\{V_{T1}, V_{T2}, \ldots V_{TT}\}$. For each ($i^{th}$) map, we find the V_T-difference value w.r.t. each of the other (j^{th}) N-1 maps: V_T-diff(i,j) = $|V_{T1}^{i}-V_{T1}^{j}|+|V_{T2}^{i}-V_{T2}^{j}|+\ldots+|V_{TT}^{i}-V_{TT}^{j}|$. Then, V_T-diff(i) is the average V_T-difference of the i^{th} map w.r.t. all other maps: V_T-diff(i)= $[\{V_T\text{-diff(i,j)}\}+\ldots\{\}]/(N\text{-}1)$: for all j≠i. Thus, V_T-diff(i) is the quantification of how different the i^{th} V_T-map is from all other maps. After computing this V_T-difference for all N V_T-maps, the V_T-map with lowest V_T-diff(i) becomes the mean V_T-map.

Given the rated frequencies for each core and the mean V_T-map for the $d \times d$ mesh, a continuous distribution of minimum core voltages can be computed (using equation 1) for the current core-to-tile mapping.

4.1.2 Formation of contiguous *VI*s

Given the continuous distribution of minimum core voltages over the 2D mesh (example shown in figure 2(a)), and set *S* of candidate supply voltage levels; in this step, we form contiguous *VI*s (figure 2(e)). Considering all candidate voltage levels in decreasing order (starting with the highest V_{dd} level), we list all the tiles with minimum voltages less than or equal to the current V_{dd} and greater than the next lower V_{dd}. We then attempt to connect all these tiles in order to form a single contiguous *VI* of the current V_{dd} level (figure 2(b)). This can create disjoint islands of tiles with minimum voltages within the current voltage range (shown as grey tiles in figure 2(c)). We then connect each island to the biggest island (of size 2 in figure 2(c)) following either a XY or YX path between the closest pair of tiles (one from each island). Here, we choose the path with the least voltage overhead (the difference between the current V_{dd} and minimum core voltages along the path). Note that the green path is chosen in figure 2(c) because it has a lower voltage overhead compared to the red path.

In certain situations, some disjoint islands, called separated islands, may not be able to connect to the biggest island because the corresponding XY and YX paths are obstructed by previously placed *VI*s of higher V_{dd} levels. In figure 2(d), the en-circled separated island cannot be connected to the biggest island (other grey tiles) to form a single contiguous *VI* of the current V_{dd} (0.9V). Here, we assign the biggest contiguous island with the current V_{dd} (green *VI* in figure 2(e)) and connect the separated islands to their nearest previously placed *VI*s (V_{dd} = 1.0V in figure 2(e)). Note that as we consider placing *VI*s in decreasing order of V_{dd} levels, all previously placed *VI*s would be of voltage levels higher than the current V_{dd}, therefore, connecting separated islands to them will not violate minimum voltage requirements of separated islands.

Finally, the minimal voltage assignment method yields a set of contiguous *VI*s on the mesh, such that the current voltage-assigned core-to-tile mapping solution satisfies the frequency constraints of all cores for the mean V_T-map.

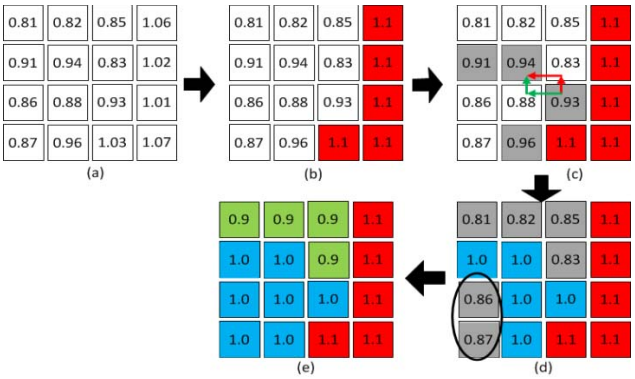


Figure 2: An illustrative example of our minimal voltage assignment: *VI*s are color-coded and annotated with V_{dd} levels, and grey tiles are marked to be connected together to form the *VI* of current V_{dd} level

4.2 SA-loop and solution perturbations

In this step, our goal is to modify the solution from the previous step to optimize PPY. We make use of a simulated annealing (SA) algorithm in this step to iteratively perturb the current solution in one of three ways, as described later in the section, with an objective function that maximizes PPY. After every solution perturbation, our minimal routing path allocation (Section 4.3) and PPY evaluation (Section 4.4) is performed. The PPY evaluation step calculates the total (computation + communication) power dissipation for all the test-chips that satisfy the performance constraints. For the current synthesis solution, the number of test-chips (out of N test-chips) satisfying both power and performance constraints is the current PPY. The SA procedure ultimately outputs the synthesized MPSoC solution with the highest PPY. The three main perturbations used in our SA-loop procedure are:

Mapping perturbation: Two cores are randomly chosen to be swapped with each other. Note that once the current mapping is perturbed, *VI*-assignment for the current solution may no longer satisfy performance constraints with respect to the mean V_T-map. Therefore, we perform minimal voltage assignment (discussed in section 4.1) after every mapping perturbation.

VI-assignment perturbation: A tile on the mesh, which is on the periphery of a *VI* and is adjacent to another *VI* is randomly chosen. Then, the V_{dd} level for the chosen tile is either hiked or lowered in order to transfer it into the neighboring *VI*. Note that any *VI*-assignment perturbation is valid only if it does not

violate the contiguity of any of the *VI*s on the mesh. Therefore, validity checks are necessary before this perturbation is executed. Figure 3 shows an example of both a valid and an invalid *VI*-assignment perturbation.

Hike highest voltage perturbation: Only when the highest V_{dd} for the current solution is lower than the highest V_{dd} in set *S*, we choose any one core from the existing highest voltage *VI* of the current solution and hike the voltage to the next higher candidate voltage level (such that this *VI* remains contiguous). Note that the *VI*-assignment perturbation does not introduce new candidate voltages from set *S*. Therefore, the hike highest voltage perturbation is occasionally applied to explore the higher candidate V_{dd} levels in the solution search-space as well.

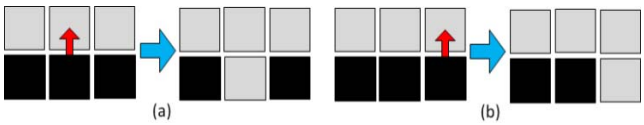


Figure 3: Illustrating valid and invalid *VI*-assignment perturbations: a tile from the black *VI* is transferred to the neighboring grey *VI*. (a) Example of an invalid perturbation, where the black *VI* no longer remains contiguous (b) Example of a valid perturbation.

4.3 Minimal routing

Given a mapped and voltage-assigned solution, our routing heuristic allocates minimal paths to all communication flows to generate a synthesized MPSoC solution. Note that whenever an inter-island link is inserted, voltage level converters (VLCs) are required in the corresponding NoC routers. These VLC components incur an overhead in terms of power dissipation and delay. Thus, the main objective of our routing step is to find a path for each communication flow such that the number of inter-island links is reduced. For each communication flow, we consider all candidate minimal paths. Out of these candidate minimal paths, we choose a routing path based on the following optimization objectives (in that order): 1) minimize total number of inter-island link insertions needed on the path; and 2) minimize total number of intra-island link insertions needed on the path.

As in [9], the communication flows are sorted in the increasing order of their path lengths, in decreasing order of their communication bandwidths for the same path length; and routed in that order. While routing any communication flow over a given path, links insertions are performed whenever the existing link(s) cannot support the bandwidth of the current flow or if no links are available. In summary, this routing scheme optimizes the number of inter-island links as well as intra-island links, thereby minimizing communication power.

4.4 PPY evaluation

Here we discuss the evaluation of performance yield (PY) as well as power-performance yield (PPY) for any given complete synthesis solution – voltage-assigned, mapped, and with routing paths allocated. The synthesis solution needs to be evaluated for satisfaction of performance and power constraints, for each of the *N* test-chips. To check if a certain test-chip satisfies the performance constraints for a given solution, we compute the maximum frequency that each of the *T* cores can be clocked at (using equation (1)). If all *T* cores can be clocked at their rated frequencies, then, the test-chip under consideration is said to satisfy performance constraints for the given synthesis solution. Thus, the number of test-chips (out of *N*) that satisfy performance constraints, becomes the performance yield for the given synthesis solution.

In order to evaluate the PPY for a given solution, we evaluate total power dissipation of only those test-chips which satisfy the performance constraints. If the total power dissipation is found to be lower than *PC*, then power constraint is considered satisfied for that test-chip. Note that routers, MCFIFOs, and VLCs operate at the same voltages/frequencies as the cores they are associated with, and are affected by the same variations in threshold voltage. Therefore, for the given synthesis solution, communication power is also evaluated for each of the *N* test-chips. In summary, the PPY evaluation step calculates the number of test-chips (out of *N*) that satisfy both performance and power goals for a given synthesis solution.

5. Experiments

5.1 Experimental setup

Our experiments were conducted using six parallel application benchmarks: four from the SPLASH-2 benchmark suite [17] (*fft*, *raytrace*, *lu*, and *cholesky*), and two from the PARSEC benchmark suite [18] (*vips* and *dedup*). Our core graphs are modeled based on the inter-core communication characterization from [19]. Each vertex in a core-graph corresponds to a producer/consumer core and the edge weights represent inter-core communication bandwidths (based on our observations of traces and communication patterns between the respective pair of cores). The rated operating core frequencies range from 600 MHz to 1300 MHz, based on the level of compute intensity of the tasks assigned to the respective cores. We use the ARM Cortex-A9 multi-core processors [20] as the baseline MPSoC compute cores, which support five operating voltage levels (S=5): 0.8V, 0.9V, 1.0V, 1.1V, and 1.2V. We use 64 core and 100 core meshes (with core-graphs of same sizes) for MPSoC platforms with dimensions 8×8 and 10×10.

Much like prior work (e.g., [15]) we use a total of 1000 test-chips (N=1000), in our experiments. The 1000 V_T-maps are generated using the open-source tool [21] (based on systematic and random WID-variation model in [22]), for mesh sizes (8×8) and (10×10). The values of 0.4 and 0.09 are used for the statistical mean and a standard deviation of the parameter V_T respectively, and the correlation range (ϕ) of 0.5 is used (as recommended in [22]). A normally distributed V_T bias, representing the D2D variation component is superimposed onto these V_T-maps; the standard deviation of the D2D V_T is assumed to be 6%, as in [23].

The power values of routers and links (32-bit wide) for different voltages, frequencies and router complexities at varying communication loads, for 45 nm technology node are obtained from ORION 2.0 [24]. Note, the router power values obtained are for nominal V_T, and are scaled for varying V_T values. Also, as all cores operate at their own rated frequencies, inter-core communication requires MCFIFOs and VLCs for inter-island communication. We consider the power overhead of these components (proportional to the voltage supply), with the actual power values based on reported overheads from existing literature [25].

In our implementation of the SA-based algorithm, we

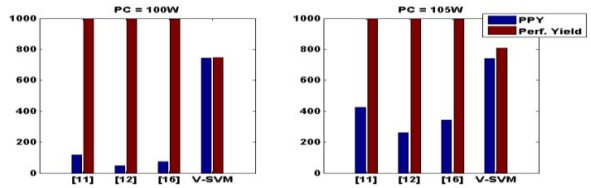
simulate for 1000 iterations (with SA temperatures between 100 and 34), after which the solution quality does not seem to improve. To effectively explore both the mapping and *VI*-assignment solution search spaces, we invoke the three perturbations with the following probabilities: *(1)* Mapping perturbation: P_1=4/40, *(2)* *VI*-assignment perturbation: P_2=34/40, and *(3)* Hike highest voltage perturbation: P_3=2/40. Two chip-wide power-constraints (*PC_stringent* and *PC_nominal*) are used in our experiments. For 64 core benchmarks: *PC* values of 100W and 105W are used, and for 100 core benchmarks: *PC* values of 150W and 160W are used.

5.2 Results

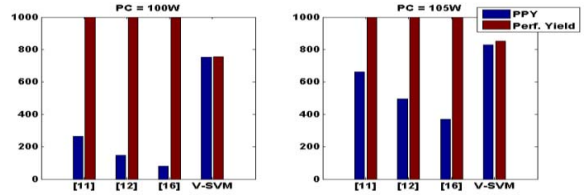
To evaluate the quality of solutions generated by our variation-aware simultaneous voltage assignment and mapping synthesis framework (hereafter referred to as *V-SVM*), we compare our results with three frameworks from prior work that we implemented along with our *V-SVM* framework.

Two of these frameworks from prior work perform process variation-aware and *VI*-aware MPSoC synthesis [11], [12]. The framework from [11] performs *VI*-assignment and core-placement to optimize the communication in the NoC, while attempting to retain the cloud-like shapes of *VI*s, in order to limit the spatial extent of systematic WID variations within individual *VI*s, thereby optimizing total energy. While [11] just considers the shapes of *VI*s, not the *VI*-locations on the mesh, the framework in [12] performs a variation-aware *VI*-placement that improves compute energy efficiency based on a single V_T-map; then core-to-tile mapping is performed within each *VI* to optimize communication energy. Both [11] and [12] do not use the knowledge of multiple variation maps, ignore chip-wide power-constraints, and optimize total power for a perfect performance yield. On the other hand, our *V-SVM* framework trades-off performance yield with power-yield for better PPY. We also compare our work against a third framework that performs variation-aware MPSoC synthesis [16]. Although, the framework in [16] does consider multiple test-chips, it does not consider *VI*s for power saving. Therefore, while implementing the approach in [16], we assume a relatively high voltage supply (1.1V) for all cores, such that almost all test-chips can satisfy all frequency constraints. Also, as [16] does not discuss the routing mechanism used, we use our minimal routing scheme in our implementation of [16] for an objective comparison with our *V-SVM* framework. We implemented the frameworks in [11], [12], and [16] to the best of our understanding and used the same power and variation models for all implemented approaches to ensure a fair comparison of the algorithms used.

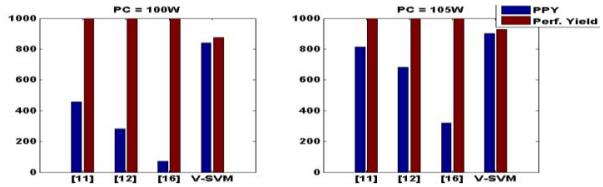
Figure 4 shows a comparison of the results obtained from using the four frameworks, in terms of performance yield and power-performance yield (PPY). Results for benchmarks with 64 core implementations are shown in figure 4(a)-(c) and for benchmarks with 100 core implementations are shown in figure 4(d)-(f)). Notice that for all the frameworks, the PPY increases as *PC* is relaxed (from left to right for any benchmark). Also note that as the primary objective of *V-SVM* is to trade-off performance yield with power-yield, the resulting PPY is generally very close to performance yield. On the other hand, for all other frameworks, even though the synthesis solutions achieve almost perfect performance yields, their PPY is quite poor.



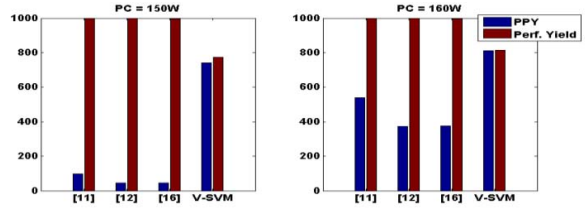
(a) SPLASH2-*fft*_64



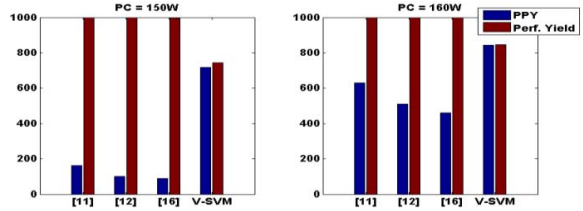
(b) SPLASH2-*raytrace*_64



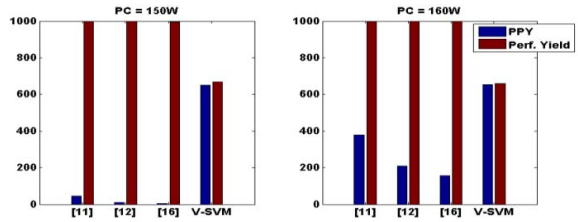
(c) PARSEC-*dedup*_64



(d) SPLASH2-*lu*_100



(e) SPLASH2-*cholesky*_100



(f) PARSEC-*vips*_100

Figure 4: Yield comparison for all four synthesis frameworks for 64 core and 100 core MPSoCs, using applications from SPLASH2 and PARSEC benchmark suites. PPY and performance yield are evaluated for each benchmark, using two power constraint (*PC*) values.

Both [11] and [12] assume a single WC V_T-value for all T tiles while performing voltage-assignments of cores, whereas, *V-SVM* combines the *VI*-assignment and mapping search-spaces to perform more power-efficient voltage assignments than [11] or [12], based on the knowledge of chip-wide *PC* as well as performance constraints corresponding to the target set of V_T-maps. Our *V-SVM* framework considers *VI*s with multiple supply voltages, which enables it to take advantage of a varied distribution of core frequencies to optimize total power. In contrast, by using a single voltage supply level for all cores, the optimization search space in [16] is quite limited, resulting in poor PPYs.

The framework from [11] outperforms [12] because [12] performs core/*VI* placement assuming only a single V_T-map, whereas [11] considers average trends during cloud shaped *VI* placement which provides better results across multiple V_T-maps. Both [11] and [12] outperform [16] as [16] does not use *VI*s (both [11] and [12] use *VI*s), which results in [11] and [12] producing lower voltage assignments that lead to higher PPYs.

Figure 4 also shows that *V-SVM* framework yields better improvements when the power constraint is more stringent. This is because a stringent power constraint lends more opportunity for our SA-loop to search for solutions with a better trade-off between performance and power. Such a trend can be readily observed in all benchmarks, by comparing the improvements obtained with *V-SVM* for different *PC* values. We can also observe a similar trend across benchmarks. For example, for a benchmark with low average power dissipation (SPLASH2-*cholesky*_100) where the given *PC* values are somewhat lax, gains in PPY with *V-SVM* (over [11], [12], and [16]) are smaller compared to gains for a communication intensive, high power dissipation benchmark (PARSEC-*vips*_100) where the same *PC* values become quite stringent.

In summary, when PPYs are averaged across all 6 benchmarks and 2 PCs (i.e., across a total of 12 cases), *V-SVM* produces average PPY improvements of 2×, 2.9×, and 3.8× compared to the frameworks proposed in [11], [12], and [16] respectively. Specifically, the improvements are 1.4×, 1.9×, and 2.4× with *PC_nominal*, and 3.9×, 7×, and 12× with *PC_stringent* over [11], [12], and [16]. On the other hand, average performance yield obtained from using *V-SVM* is 0.79×, 0.79×, and 0.8× over that obtained from the frameworks in [11], [12], and [16] respectively. Thus, rather than just aggressively optimize performance yield as done in prior work, our proposed *V-SVM* framework trades-off performance with power for a more desirable power-performance yield.

6. Conclusion

Due to the considerable impact of process variations in modern technologies, variation-awareness is essential even at the system-level design stage. Traditional design flows where voltage assignments are done without the core-mapping information require resorting to pessimistic voltage assignments based on worst case V_T values. This leads to higher voltage assignments and a corresponding higher overall power dissipation. For the first time, we propose a variation-aware system-level synthesis framework for simultaneous voltage assignment and mapping in MPSoCs with *VI*s. Our framework incorporates novel *VI*-aware and variation-aware optimization techniques, and produces on average 2× to 3.8× improvements in power-performance yield (PPY) over other variation-aware MPSoC synthesis frameworks. Such a framework has immense utility for emerging MPSoC designs.

Acknowledgement

This research is sponsored in part by grants from NSF (CCF-1252500), SRC, and AFOSR (FA9550-13-1-0110).

References

[1] E. Humenay, D. Tarjan, K. Skadron, "Impact of process variations on multicore performance symmetry," Proc. DATE 2007, pp. 1653-1658.

[2] L. Pang et al. "Measurement and Analysis of Variability in 45 nm Strained-Si CMOS Technology," IEEE JSSC, Aug. 2009.

[3] S. Reda, and S. Nassif, "Analyzing the impact of process variations on parametric measurements: novel models and applications," Proc. DATE, 2009, pp. 375-380.

[4] S. Garg, and D. Marculescu, "System-level process variation driven throughput analysis for single and multiple voltage-frequency island designs," Proc. DATE, 2007, pp. 1-6.

[5] S. Vangal et al., "An 80-Tile 1.28 TFLOPS Network-on-Chip in 65 nm CMOS," IEEE ISSCC, 2007, pp. 98–589.

[6] Tilera Corporation. TILE64™ Processor. Product Brief. 2007.

[7] U. Ogras, et al., "Voltage-frequency island partitioning for GALS-based networks-on-chip," Proc. DAC, 2007, pp. 110-115.

[8] W. Jang, D. Ding, D. Pan, "A voltage-frequency island aware energy optimization framework for networks-on-chip," Proc. ICCAD, 2008.

[9] N. Kapadia, S. Pasricha, "A Framework for Low Power Synthesis of Interconnection Networks-on-Chip with Multiple Voltage Island," Integration, the VLSI Journal, vol. 45, no. 3, pp. 271-281, June. 2012.

[10] Intel SCC, http://techresearch.intel.com/, 2010.

[11] S. Majzoub, R. Saleh, S. Wilton, and R. Ward, "Energy Optimization for Many-Core Platforms: Communication and PVT Aware Voltage-Island Formation and Voltage Selection Algorithm", IEEE Trans. Computer-Aided Design, vol. 29, no. 5, pp. 816-829, May 2010.

[12] N. Kapadia and S. Pasricha, "VERVE: A Framework for Variation-Aware Energy Efficient Synthesis of NoC-based MPSoCs with Voltage islands," ISQED, 2013, pp. 603-610.

[13] F. Wang et al., "Variation-Aware Task and Communication Mapping for MPSoC Architecture," IEEE Trans. CAD, vol. 30, no. 2, Feb. 2011.

[14] D. Mirzoyan, B. Akesson, and K. Goossens, "Process-variation aware mapping of real-time streaming applications to MPSoCs for improved yield," Proc. ISQED, 2012, pp. 41-48.

[15] L. Huang and Q. Xu, "Performance Yield-driven Task Allocation and Scheduling for MPSoCs under Process Variation," Proc. DAC 2010.

[16] K. Bhardwaj, S. Roy, and K. Chakraborty, "Power-Performance Yield Optimization for MPSoCs Using MILP," Proc. ISQED, 2012.

[17] S.V. Woo, M. Ohara, E. Torrie, J.P. Singh, and A. Gupta, "The SPLASH-2 programs: characterization and methodological characterization," Proc. ISCA, pp. 24-36, May 1995.

[18] C. Bienia, S. Kumar, J.P. Singh, K. Li, "The PARSEC benchmark suite: characterization and architectural implications," Proc. PACT, 2008.

[19] N. Barrow-Williams, C. Fensch, and S. Moore, "A communication characterization of Splash-2 and Parsec," IEEE IISWC, pp.86-97, 2009.

[20] ARM, http://www.arm.com/products/processors/selector.php

[21] VARIUS model, http://www.cse.ohiostate.edu/~teodores/arch/tools/

[22] S. Sarangi et al., "VARIUS: A Model of Process Variation and Resulting Timing Errors for Microarchitects," IEEE Trans. Semiconductor Manuf., vol. 21, no. 1, pp. 3-13, Feb. 2008.

[23] B. Li, L. Peh, and P. Patra, "Impact of process and temperature variation on network-on-chip design exploration," NOCS, 2008.

[24] A. Kahng, B. Li, L. Peh, and K. Samadi, "ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration," Proc. DATE, 2009, pp. 423-428.

[25] T. Chelcea, S. Nowick, "A low-latency for mixed-clock systems," CSW, pp. 119–126, Apr. 2002.

[26] N. Kapadia, S. Pasricha, "VISION: A framework for voltage island aware synthesis of interconnection networks-on-chip," Proc. GLSVLSI, 2011, pp. 31-36.