Energy-Efficient Mapping of Biomedical Applications on Domain-Specific Accelerator under Process Variation

Mohammad Khavari Tavana†, Amey Kulkarni‡, Abbas Rahimi◊,
Tinoosh Mohsenin‡ and Houman Homayoun†
†Department of Electrical and Computer Engineering, George Mason University, Fairfax County
‡ Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County
◊Department of Computer Science and Engineering, UC San Diego, La Jolla
Email: †{mkhavari, hhomayou}@gmu.edu ‡{ameyk1,tinoosh}@umbc.edu ◊abaas@cs.ucsd.edu

ABSTRACT

The variability of deep-submicron technologies creates systems with asymmetric cores from a frequency and leakage power viewpoint, which makes an opportunity for performance-power optimization. In particular, process variation can transform a homogeneous many-core platform into a heterogeneous system where the task mapping becomes extremely difficult. In this paper, we propose a mapping algorithm that selects an appropriate task mapping along with voltage and frequency assignment for a cluster of cores. The mapping algorithm, which is based on simulated annealing, determines cluster voltages and core frequencies to minimize energy consumption and EDP under process variation. We examine the effectiveness of our proposed algorithm on a fully placed and routed 128-core biomedical accelerator in 45nm when running various applications including compressive sensing, seizure detection and ultrasound spectral Doppler and linear regression. The results indicate that exposing frequency and power variations to the mapping algorithm results in up to 22% (on average 11%) energy saving and 31% (on average19%) EDP improvement.

Categories and Subject Descriptors

C.3 [**Special-purpose and application-based systems**]: Real-time systems and embedded systems

General Terms: Algorithms, Design

Keywords: Mapping, accelerator, process variation, many-core systems, energy efficiency

1. INTRODUCTION

Unsustainable power consumption and ever-increasing computing demands have driven the computing industry to move to an era of parallelization with few to tens of computing cores integrated in a single die. Domain-specific customization such as programmable many-core accelerator has been emerged as the next disruptive technology to bring significant performance and power-efficiency improvement [1]. The International Technology Roadmap for Semiconductors (ITRS) predicts that single embedded SoCs will have tens of specialized accelerators by 2021 [2]. With embedded devices being battery powered, energy efficiency of this class of architectures becomes a major concern. In particular, as the number of cores increases in many-core accelerator the power dissipation increases and therefore limits the scalability of this class of design [3].

Among conventional low power design techniques, dynamic voltage/frequency scaling (DVFS) is one of the most effective ones. This technique has been extensively explored in managing the balance between power and performance in multi-core architectures [4]. However, in more recent technologies where the device dimensions approaching the limits of process technology capabilities, a rapid increase of manufacturing process variation is observed [6] [13]. These variations result in the spread of maximum achievable clock frequencies (*fmax*) across different cores in a chip where employing DVFS to save power and energy should be considered wisely. For example, the results acquired using *Synopsys PrimeTime-VX* for a 45nm technology shows 12% frequency variation at 1.1V. The variation further increases for lower operating voltages. As a result, the variability in the process parameters leads to a system with asymmetric cores from a frequency and leakage power viewpoint. Consequently, the process variation transforms a homogeneous many-core accelerator into a heterogeneous system. Hence, variation-aware task mapping and voltage/frequency assignment is required [14]. The energy-efficient mapping of tasks and voltage/frequency management necessitates specific strategies to exploit the diversity of the accelerator cores while meeting the energy and/or real-time constraints. In addition, in cluster-based accelerator architectures the processing cores are grouped into a set of clusters where a cluster is controlled by the same power supply voltage domain. This makes the task mapping strategy crucial and can have significant impact on the energy-efficiency of the system.

This paper highlights the challenges and investigates a solution for energy management in many-core accelerator systems in the presence of process variation and under real-time constraint. We utilize simulated annealing to find optimal task mapping, parallelization degree of tasks, and DVFS setting with few voltage domain clusters. The rest of the paper is organized as follows: Section 2 introduces models and assumptions, while Section 3 provides a motivational example about the task mapping. Section 4 introduces the mapping algorithm based. Section 5 is our evaluations and experimental results, and Section 6 concludes the paper.

This work was supported in part by the National Science Foundation under grant NSF CNS– 1329829.

ISLPED'14, August 11 - 13 2014, La Jolla, CA, USA
Copyright 2014 ACM 978-1-4503-2975-0/14/08$15.00.
http://dx.doi.org/10.1145/2627369.2627654

2. MODELS AND ASSUMPTIONS

2.1 Architecture Model

We focus on an accelerator comprised N_{core} homogeneous cores, each of which can process its data and execute its own instruction streams through I/O ports and local memories. The cores are equipped with DVFS [4] to save power/energy consumption. Even though it is possible to assign each core a dedicated voltage DC-DC convertor, it is not efficient due to the complexity of chip design

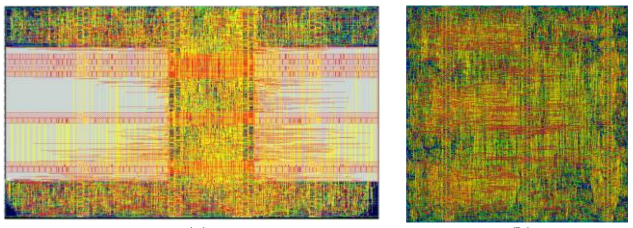

Fig. 1: (a) Core layout and (b) router layout. The core and router area footprint are 0.033 mm^2 and 0.035 mm^2 respectively in 45nm technology, where each router is shared by 4 cores.

and area overhead. This problem exacerbates when the number of cores is in the orders of hundreds each with small area footprint [5]. As a consequence, a group of cores forms a voltage domain cluster, such that the cores on the same cluster share the same power lines and supply voltage. Our accelerator platform consists of 128 processing cores divided into 8 voltage clusters where each cluster contains 16 cores.

The cores are based on a 6-stage modified RISC pipeline architecture which communicate through a simple, scalable hierarchical 4-ary tree structure that reduces the number of hops in communication. The cores are equipped with parallel loop control, parallel FFT processing and pointers to accelerate computation [11]. Note that each core is quite small and contains dedicated 128 words data and instruction memories. Hence, if there is not enough memory space to hold all the required data and instruction needed for a task execution, the data should be distributed among cores. Each core and router were synthesized, placed and routed in a 45nm CMOS process (Fig. 1). Each core operates at different clock rates through Globally Asynchronous Locally Synchronous (GALS) architecture [10] thereby eliminating global clock routing.

2.2 Application Model

We consider two cases: i) real-time applications where all tasks share a common deadline *D*, i.e., the application deadline. In this case, we aim at minimizing energy consumption provided that application is finished before the deadline. ii) non real-time applications where we aim at minimizing energy-delay product. Each task is either *serial* or *moldable*. In the former case a task cannot be parallelized, but in the latter the parallelization degree of a task can be determined before mapping and application execution. We assume that there is a control or data dependency among tasks in our applications which is modeled by direct acyclic graph (DAG). A sample application can be represented as $TG = (V, E, C, T)$. The first two parameters shows the topology of graph where $n_i \in V$ represents a set of nodes (i.e., tasks) and E indicates their edges (i.e., dependency among tasks). For example $(n_i, n_j) \in E$ shows there is an edge between task *i* and *j* and the latter cannot be executed before the former one due to the dependency. $c_{ij} \in C$ indicates the communication cost between node *i* and *j* in terms of number of flits, and $T_i^p \in T$ indicates execution cycles of task *i* with *p* level of parallelism. For serial tasks, *p* is one, but for moldable tasks *p* varies from one to the maximum level of parallelism which is also task dependent. Note that parallelizing tasks may impose new subtasks in the task graph, and also increases the amount of communications.

2.3 Process Variation and Power Models

Because of the imperfection during manufacturing process, the maximum frequency and the static power dissipation will be varied from one core to another. To find the effect of process variation on the frequency, the RTL description of the fabric has been synthesized for 45nm technology, the general purpose process. The front-end flow with multi-VTH cells has been performed using *Synopsys Design Compiler* with tight timing constraint, while *Synopsys IC Compiler* has been used for the back-end. First, to observe the effect of static process variation on the fabric, we have analyzed how the critical paths of each core are affected due to within-die and die-to-die process parameters variation. Therefore, the various cores within the fabric experience different variability-induced delay and thus display various error rate. During the sign-off stage, we have injected process variation in the fabric using the variation-aware timing analysis engine of *Synopsys PrimeTime-VX*. Fig. 2 show the frequency variation of the first sixteen cores of the 128 core accelerator platform. At three higher voltage levels almost 12% variation is observed in frequency, and frequency variation exceeds 15% at the lowest voltage level. We performed the frequency binning with the step of 33 MHz for the accelerator, hence, the frequency of each core varies from 366 MHz up to 933 MHz. The dashed line in Fig. 2 indicates 633 MHz frequency. Note that only nine out of 16 cores can operate at this frequency at the voltage of 0.81V. Moreover, four cores are operational below 600 MHz at this voltage level.

Table 1: power consumption of the processing core

Operating Voltage	Freq. (MHz)	Dynamic Power(mW)	Static Power(mW)	Total Power(mW)
1.10V	*900*	*141.39*	*3.90*	*145.29*
0.99V	*733*	*62.63*	*2.12*	*64.75*
0.81V	*633*	*34.06*	*1.14*	*35.20*
0.66V	*433*	*18.96*	*0.99*	*19.95*

The energy dissipation comprises of dynamic and static energy consumption. The former is due to the activity and switching of gates in the system, and the latter is the amount of energy dissipated when the system is in the standby state. In Table 1, the power breakdown of a core at different voltage levels has been presented. The static power variation has been modeled similar to [7], moreover, for communication energy modeling we used the modeled provided by [12]. It is noteworthy that it takes 4 cycles to send a flit from router to router, or core to router in our system.

3. MOTIVATIONAL EXAMPLE

To evaluate how mapping strategy in the cluster-based accelerator and under process variation can influence energy consumption, we provide a simple example in which an application with three tasks is mapped on a system with four cores. Even though we consider dependency in our example, for simpilicity, we assume there is no communication among the tasks. The system is divided into two clusters each composed of two cores and can operate at two different voltages/frequencies. The goal of mapping is to assign tasks to the cores and select the voltage and frequency to minimize the energy consumption provided that the applicaion meets the

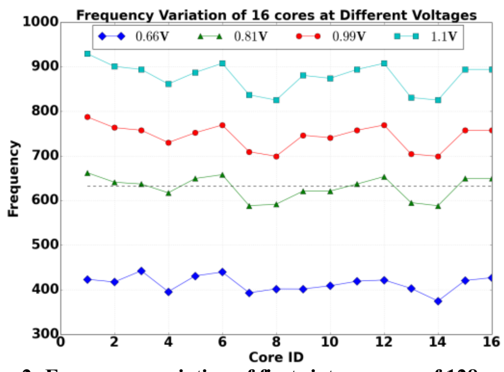

Fig. 2: Frequency variation of first sixteen cores of 128-core accelerator obtained by *Prime Time-VX* for a 45nm technology

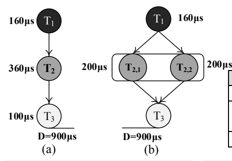

Table 2

Task	Execution Time
T_1	160 μs
T_2	with $p=1 \rightarrow 360\ \mu s$
	with $p=2 \rightarrow 200\ \mu s$
T_3	100 μs

Table 3

CoreID	Max Frequency (MHz)		Leakage Power(mW)	
	1 V	0.6 V	1 V	0.6 V
1	800	550	8	3
2	750	450	5	2
3	700	400	2	1
4	650	350	3	1.5

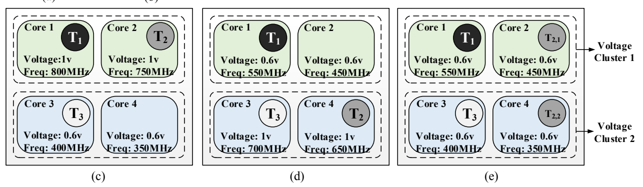

Fig. 3: Mapping an application on a 4-core system with different leakage power and frequency characteristics.

deadline. For each core the maximum energy saving at a given voltage is achieved by executing the task at the maximum frequency that can be supported .Therefore, whenever we determine a voltage level, we set the operating frequency at the maximum possible corresponding to that voltage. Table 2 shows tasks execution time, and Table 3 represents the maximum frequency at a given voltage and the leakage power for each core. It is important to note that T_2 can be executed either as a single task (Fig. 3 (a)) with 360 μs execution time or as two sub-tasks each with 200 μs execution time (Fig. 3 (b)). We assume that the execution time of each task is obtained through profiling on a core clocking at 800 MHz. Note that in the presence of process variation the operating frequency might be lower or higher than the profiling frequency and therefore the mapping algorithm need to be adapted accordingly, to meet the deadline. In addition, we assume 50 mW dynamic power is dissipated when each processing core operates at the maximum voltage level, and it decreases quadratically with reducing voltage level linearly [4].

Fig. 3 (c) represents a mapping where tasks T_1, T_2, T_3 are mapped on cores 1, 2 and 3 respectively. None of the tasks are mapped on core 4 because it is the weakest core among the others. Note that due to the mapping of T_2 on the core 2, the cluster 1 cannot reduce the voltage, because it leads to application deadline violation, while cores on cluster 2 can use lower voltage level to reduce energy. If we consider perfect power gating for core 4, 43.4 μj energy is consumed with this mapping. By moving task T_2 from core 2 to core 4 (i.e., the weakest core), the voltage of the first cluster can be reduced while the second cluster cannot. This case is shown in Fig. 3 (d) which results in 39.25 μj energy consumption, and consequently 10% energy saving compared to the first mapping. In Fig. 3 (e), we show how exploiting parallelism of tasks can help to increase the system utilization and further improve energy efficiency. Although breaking task T_2 into two sub-tasks lead to sub-linear speedup in our case, it allows both clusters to operate at lower voltage without violating application deadline. Mapping two sub-tasks $T_{2,1}$ and $T_{2,2}$ on cores 2 and 4 respectively lead to 29.16 μj and additional 26% energy saving compared to the mapping in Fig. 3 (d).

Mapping tasks on many-core platforms such as an accelerator where there is a variation on power and frequency is similar to mapping problem to a heterogeneous hardware platforms which is NP-hard in a strong sense [8]. The search space of the mapping problem even gets larger, assuming that tasks can be parallelized and the tasks set parameters like execution time can be changed dynamically.

4. MAPPING ALGORITHM

Simulated annealing (SA) heuristic is a popular approach for solving both discrete and continuous optimization problems and can effectively deal with nonlinear and multivariate systems [9]. Our energy-efficient mapping algorithm is based on this heuristic technique. This algorithm probabilistically allows poorer solutions to be accepted to better search the solution space and move out from local optima. A standard kind of SA-based algorithm can be found in [9], the parameters and how the new solutions are generated form the fundamental parts of this algorithm which are case dependent [9]. The algorithm begins with a random feasible mapping as the initial solution and progresses with new movements from the current state.

Determining the level of parallelism for moldable task not only has an impact on the energy consumption for a task itself, but also influences on the efficiency of mapping for the entire tasks set. Two movements have been employed in the algorithm, which allows the level of parallelism for each task to be expanded or shrunk.

- *Increasing parallelism:* randomly selects a moldable task and increases the level of parallelism for a given task. If increasing violates the upper bound of the parallelism, the action will be discarded.
- *Decreasing parallelism:* randomly selects a task and decreases the level of parallelism for a given task. If decreasing leads to elimination of the task in the mapping, the action will be discarded.

Apart from expanding or shrinking, we have also used four moves to alter mapping. It is possible that these small alternations in the mapping lead to modification of clusters voltage and as a consequence result in significant change in the energy dissipation.

- *Intra-cluster task swapping:* two random cores within a cluster are selected and their corresponding tasks are swapped together. It is possible that one of the core to be totally available (free), hence in this case we have simple task movement rather than swapping. In case where both selected cores are free, the movement is discarded.
- *Inter-cluster task swapping:* two random processing cores among the clusters are selected and their corresponding tasks are swapped together. It is possible that one of the core to be totally free. Therefore, in this case we can have simple task movement rather than swapping. In case both selected core are free, the movement is discarded.
- *Cluster swapping:* two random clusters and their tasks are swapped together. In case both selected clusters are totally free the movement is discarded. Even though this movement can be render with many inter-cluster tasks swapping, our simulations showed that adding this movement to our solution generator procedure can increase speed of the algorithm and quality of the solution.
- *Random movement:* although three former movements are systematic, adding some flavor of randomness can help the algorithm to better explore the search space and avoid from sticking at the local optima.

If any movement violate the timing constraint of the application, it will be discarded. Notice that in case of discarding any movement, the algorithm regenerates solution once again. For a given voltage, the maximum operational frequency of the processing core leads to minimum energy consumption. As the voltage is controlled at the cluster level, the minimum energy dissipation for a cluster can be found by simply checking a small set of voltage levels. The bad mapping can dramatically affect the flexibility of voltage reduction because the tightest time constraint of tasks in a cluster determines the amount of voltage/frequency reduction. We defined Δ as the

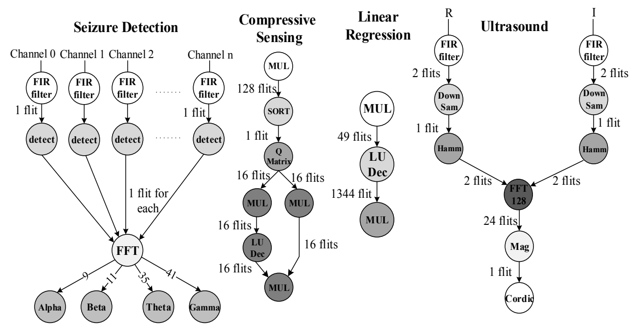

Fig. 4: task graph of our biomedical applications

difference between energy consumption (or EDP) of new mapping and current mapping. The energy consumption includes both cores energy and communication energy.

If Δ becomes negative it means that generated solution is the better one and replaced with the current solution. Otherwise, the solution is accepted with probability of $e^{-\Delta/T}$, where T is the current temperature. When T is larger, the probability of accepting worse solution is more and by reducing T this probability reduces.

Whenever the neighbor solutions around optimal one are much closer to each other, the optimal solution hardly froze even at a very low temperature. Hence, we set a bound for minimum temperature in which crossing that will stop the algorithm. This bound is small enough (0.001) which allows the solution to be frozen most of the time.

After the predefined number of iterations, the temperature should be reducing by *cooling rate* parameter. We used exponential cooling scheme [9], which is the most common cooling decrement scheme. We set the *cooling rate* equal to 0.92, *initial temperature* equal to 60, and *initial iteration* equal to 600 in our simulations.

5. SIMULATION FRAMEWORK

For application profiling, each algorithm is partitioned into multiple tasks where each task is assigned to a single or multiple cores. Then using our many-core simulator, different application statistics are obtained. For moldable tasks, different level of parallelism is considered and investigated to analyze the execution time and communication overhead. Because the accelerator used for simulations were designed to execute biomedical domain applications effectively [11], we provide four implementation of biomedical applications for the case study (Fig. 4) including compressive sensing, seizure detection, linear regression and ultrasound. To increase the utilization of the accelerator and measure the energy and EDP, we did the mapping with four copies of each application, except the seizure detection for which we used four channels. For the energy minimization of real-time tasks we considered application deadline equal to summation of tasks execution in serial form, thus the slack time can be created by parallelizing moldable tasks and give the opportunity to reduce voltage and frequency. In Fig. 5 (a) and (b), we compared our mapping with two other mapping scenarios, the process variation agnostic scheme where the mapping scheme is not aware of the process variation, and best random scheme where 10000 feasible random mapping were generated and the best one were considered as the solution. The energy consumption reduces up to 22% and on average 11% over process variation agnostic scheme. The saving increases up to 53% and on average 40%, respectively compared to the best random mapping. The improvement of EDP is more significant. The EDP reduces up to 31% and on average 19% over the process variation agnostic mapping. The improvement

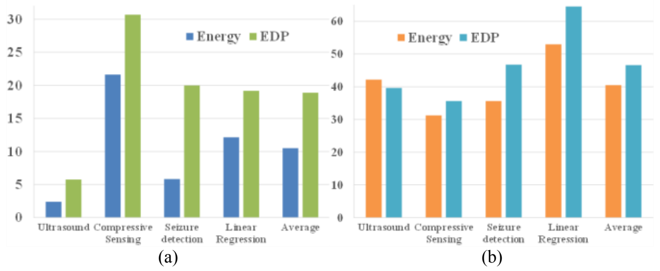

Fig. 5: (a) proposed mapping compared to process variation agnostic mapping, (b) proposed mapping compared to best random mapping

increases up to 65% and on average 47%, respectively compared to the best random mapping.

6. CONCLUSION

This paper proposes a mapping algorithm based on simulated annealing to render energy efficiency in a many-core accelerator architecture under the process variation. Leveraging the frequency and power variations in the mapping algorithm as well as the level of parallelism of the tasks, we significantly reduce the energy consumption. Energy-efficient mapping of the tasks necessitates specific strategies to exploit the diversity of the accelerator cores while meeting the application real-time constraints. The proposed mapping methodology is evaluated by biomedical applications executing on a 128-core accelerator divided into 8 voltage clusters where each cluster contains 16 cores. The results indicate that exposing frequency and power variations to the mapping algorithm results in up to 22% energy saving and 53% EDP improvement.

7. REFERENCES

[1] P. Schaumont and I. Verbauwhede, "Domain-specific codesign for embedded security." Computer, 36(4), pp. 68-74, 2003.

[2] "Semiconductor industry association, international technology roadmap for semiconductors (itrs), 2010, update 2011. [Online]. Available: http://www.itrs.net.

[3] S. Nilakantan, S. Battle and M. Hempstead, "Metrics for Early-Stage Modeling of Many-Accelerator Architectures." *Computer Architecture Letters*, 12(1), pp. 25-28, 2013.

[4] M. T. Schmitz, B. Al-Hashimi, and P. Eles. "*System-level design techniques for energy-efficient embedded systems*" (Vol. 4). Dordrecht: Kluwer Academic Publishers, 2004.

[5] S. Vangal, et al. "An 80-tile 1.28 TFLOPS network-on-chip in 65nm CMOS." *IEEE International Solid-State Circuits Conference*, Digest of Technical Papers, pp. 98-589, 2007.

[6] S. R. Sarangi, B. Greskamp, R. Teodorescu, J. Nakano, A. Tiwari, A. and J. Torrellas. "VARIUS: A model of process variation and resulting timing errors for microarchitects." *IEEE Transactions on Semiconductor Manufacturing*, 21(1), pp. 3-13, 2008.

[7] B. Raghunathan, Y. Turakhia, S. Garg and D. Marculescu. "Cherry-picking: exploiting process variations in dark-silicon homogeneous chip multi-processors." *In Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 39-44, 2013.

[8] A. Schranzhofer, J. J. Chen and L. Thiele, "Dynamic power-aware mapping of applications onto heterogeneous mpsoc platforms." *IEEE Transactions on Industrial Informatics*, 6(4), pp. 692-707, 2010.

[9] J. W. Chinneck, "*Practical optimization: A gentle introduction*." 2004, Electronic document: http://www. sce. carleton. ca/faculty/chinneck/po. html.

[10] D. Chapiro, "*Globally-asynchronous locally-synchronous systems*." Ph.D. thesis, Stanford University, 1984.

[11] J. Bisasky, H. Homayoun, F. Yazdani, and T. Mohsenin, "A 64-core platform for biomedical signal processing." *In International Symposium on Quality Electronic Design (ISQED)*, pp. 368-372, 2013.

[12] J. Hu and R. Marculescu. "Energy-aware communication and task scheduling for network-on-chip architectures under real-time constraints." *In Design, Automation and Test in Europe Conference and Exhibition*, vol. 1, pp. 234-239, 2004.

[13] S. Avesta, H. Homayoun, A. Eltawil, and F. Kurdahi. "Process variation aware sram/cache for aggressive voltage-frequency scaling." *In Design, Automation & Test in Europe Conference & Exhibition,* DATE'09, pp. 911-916., 2009.

[14] A. Rahimi, A. Marongiu, P. Burgio, R. K. Gupta, L. Benini, "Variation-tolerant OpenMP Tasking on Tightly-coupled Processor Clusters" Proc. ACM/IEEE DATE, 2013, pp. 541-546.