Combined DVFS and Mapping Exploration for Lifetime and Soft-Error Susceptibility Improvement in MPSoCs

A. Das, A. Kumar, B. Veeravalli
Dept. of Electrical and Computer Engineering
National University of Singapore, Singapore
{akdas|akash|elebv}@nus.edu.sg

C. Bolchini, A. Miele
Dip. Elettronica, Informazione e Bioingegneria
Politecnico di Milano, Milano - Italy
{bolchini|miele}@elet.polimi.it

***Abstract*—Energy and reliability optimization are two of the most critical objectives for the synthesis of multiprocessor systems-on-chip (MPSoCs). Task mapping has shown significant promise as a low cost solution in achieving these objectives as standalone or in tandem as well. This paper proposes a multi-objective design space exploration to determine the mapping of tasks of an application on a multiprocessor system and voltage/frequency level of each tasks (exploiting the DVFS capabilities of modern processors) such that the reliability of the platform is improved while fulfilling the energy budget and the performance constraint set by system designers. In this respect, the reliability of a given MPSoC platform incorporates not only the impact of voltage and frequency on the aging of the processors (wear-out effect) but also on the susceptibility to soft-errors – a joint consideration missing in all existing works in this domain. Further, the proposed exploration also incorporates soft-error tolerance by selective replication of tasks, making the proposed approach an interesting blend of *reactive* and *proactive* fault-tolerance. The combined objective of minimizing core aging together with the susceptibility to transient faults under a given performance/energy budget is solved by using a *multi-objective genetic algorithm* exploiting tasks' mapping, DVFS and selective replication as tuning knobs. Experiments conducted with real-life and synthetic application graphs clearly demonstrate the advantage of the proposed approach.**

I. Introduction

To address the growing performance demand and to meet the necessary scalability quest, multiple processing cores are interconnected together to form Multiprocessor Systems-on-Chip (MPSoCs) with an interconnection fabric (such as Networks-on-Chip, NoCs) constituting the communication backbone. Example of these systems are *OMAP* from Texas Instruments, *NEXPERIA* from Philips and *NOMADIK* from STMicroelectronics. Two of the major challenges faced by MPSoC system designers concern energy consumption and reliability of operations of the cores. These issues are aggravated with shrinking transistor geometries and growing transistor densities associated with deep sub-micron technology nodes.

Most modern cores of an MPSoC platform support a wide range of voltages and frequencies, often exploited to minimize energy consumption. In this respect, task scheduling has been used extensively, especially for MPSoC platform-based design, to scale the voltage and frequency of a core based on the workload mapped on it [1], [2]. This technique has been dubbed Dynamic Voltage and Frequency Scaling (DVFS). However, with every new technology node, the number of faults (transient, intermittent and permanent) per device is on the rise [3], and energy-aware scheduling techniques do not address fault-tolerance, that needs to be specifically dealt with. There are two key issues related to reliability – core aging (resulting in permanent faults) and soft-error susceptibility (resulting in transient faults). As established in [4], [5], the lifetime of a core is strongly dependent on its temperature, which in turn depends on the voltage/frequency of operation. Specifically, higher voltage and frequency increase the temperature, and therefore core aging. Recent studies on the causes of transient faults have shown that the soft-error susceptibility of a circuit increases by several orders of magnitude when operating at lower voltage/frequency [6], [7]. Clearly, the two objectives (core aging and soft-error susceptibility) present contradictory requirements in terms of voltage and frequency levels. Furthermore, operating a core at lower voltage and frequency results in significant energy savings but degrades performance. In this perspective it is worth mentioning that embedded systems are usually constrained in terms of a given energy budget and a minimum performance constraint. Finally, fault-tolerance techniques can be adopted for mitigating the effects of faults, especially transient; however, the introduced redundant computations may cause an additional stress implying a negative effect on the architecture aging.

Existing works have mainly tackled one aspect of the complex scenario, by acting on DVFS and mapping for optimizing the trade-off between energy and soft-error susceptibility [7], [8], possibly exploiting fault tolerant techniques [9], or by accurately mapping the application to improve the architecture lifetime [4], [5].

Contributions: This paper proposes a multi-objective optimization approach to determine the mapping $\times$ voltage/frequency level for each task of an application on a target architecture to mitigate cores' aging and minimize their soft-error susceptibility. The key contributions of this paper are the following ones:

1. *Proactive Fault-tolerance (Transient + Permanent):* Considering the effect of voltage/frequency scaling on the aging of cores and its susceptibility to transient faults.
2. *Reactive Fault-Tolerance (Transient + Permanent):* Selective task replication to further minimize the suscepti-

978-3-9815370-2-4/DATE14/©2014 EDAA

bility to transient faults and at the same time limiting the additional stress on the cores causing a faster aging.

3. A multi-objective Design-Space Exploration (DSE), and a companion automated engine based on genetic algorithms, to solve the three above-mentioned problems.

The rest of this paper is organized as follows. A brief overview of the existing research works is provided in Section II. This is followed by an introduction to the system model in Sections III and the problem formulation in Section IV. A brief description of the proposed framework and the DSE engine is presented in Section V. Experimental results are provided in Section VI and conclusions in Section VII.

II. Related Work

Mapping of applications on a multiprocessor systems is an NP-hard problem. Several heuristics have been proposed in literature to solve both the unconstrained optimization problem (best effort) as well as its constrained counterpart (such as energy and fault-tolerance). Providing an exhaustive review of all these techniques is beyond the scope of this paper. Instead, a few significant contributions are highlighted and distinctions are made with respect to the propose one.

There are quite a few studies on DSE for transient fault-tolerance. For instance, a design optimization heuristic is proposed in [9] to select between task re-execution (selective replication) and hardening of hardware to maximize the system reliability, minimizing system cost and satisfying the deadline requirement. A reliability optimization technique is proposed in [10] where the reliability requirements such as replication are embedded in the application task graph, then scheduled on a given MPSoC platform. These approaches do not consider processors' aging.

Permanent faults in terms of wear-outs have received significant research attention in recent years. A simulated annealing based approach [4] and a convex optimization based approach [5] are proposed to maximize system lifetime. None of these approaches deal with transient faults or act on DVFS.

Recently, there are techniques to simultaneously deal with transient and permanent faults. A redundancy based scheduling is proposed in [11] to maximize the reliability due to transient and permanent faults; however, aging of processors is not considered. A lifetime reliability-aware transient fault-tolerant technique is proposed in [12]. Both these techniques are energy-agnostic, an aspect that is not so secondary any more.

From the soft-error susceptibility point of view, substantial studies have been conducted to optimize for transient faults occurrence simultaneously minimizing energy consumption. An adaptive checkpointing technique is proposed in [8] to schedule the checkpoints in an application to maximize soft-error reliability, simultaneously minimizing energy consumption. A shared recovery technique is proposed in [7] to optimally use DVFS to minimize energy while simultaneously satisfying a given reliability goal. Another similar approach, but using selective replication is proposed in [13]. Finally, a Monte-Carlo simulation-based approach is proposed in [14] to estimate the task execution time; based on this, a low power and early deadline first scheduling is proposed to compensate for soft-errors.

This analysis shows that there are no existing techniques that deal with aging and transient-fault tolerance simultaneously with DVFS, a joint objective targeted in this work. Furthermore, the paper also combines proactive and reactive tolerance, not being considered in the existing works.

III. System Model

A. Architecture and Application Model

The adopted architecture model, shown in Figure 1, is composed of a set of processing cores interconnected in a mesh-based topology. Such an architecture is represented as a graph $\mathcal{G}_{arc} = (\mathcal{V}_{arc}, \mathcal{E}_{arc})$, where $\mathcal{V}_{arc}$ is the set of nodes n_j representing the processing cores and $\mathcal{E}_{arc}$ is the set of edges representing the communication channels among the cores.

As common practice (e.g., [9], [10], [13]), an application (or a set of applications) is modeled as directed acyclic graph $\mathcal{G}_{appl} = (\mathcal{V}, \mathcal{E})$ (refer to the example in Figure 2), where $\mathcal{V}$ is the set of nodes representing tasks of the application and $\mathcal{E}$ is the set of edges $\{e_{i,j} \mid 1 \leq i, j \leq |\mathcal{V}|\}$, representing data dependency among tasks v_i and v_j. Each task $v_i \in \mathcal{V}$ is annotated with its execution time τ_i (at the maximum frequency of operation, as discussed later), while each edge with the amount of transmitted data $d_{i,j}$. Transmission times depend on the size of the data and on the communication channel speed; however, for sake of space, this aspect will not be discussed in the paper.

The application $\mathcal{G}_{appl}$ is synthesized on the target architecture $\mathcal{G}_{arc}$, by *mapping* each task v_i on a specific core n_j, and *scheduling* the start time; in this model, a non-preemptive scheduling is used. It is to be noted that, each core can execute at most a task per time, and, in the execution order, data dependencies have to be satisfied. Thus, the execution period of the synthesized system is computed as the difference between the end time of the last executed task and the start time of the first executed one.

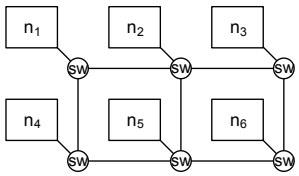

Figure 1. Architecture model.

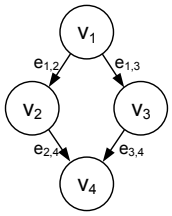

Figure 2. Application model.

B. Dynamic Voltage and Frequency Scaling

As in [1], [2], [8], each core supports a set of voltage/frequency levels in the range $[V_{min}/\omega_{min}, V_{max}/\omega_{max}]$, and each task of a given application can be associated with a specific level that is applied to the core devoted to its execution. According to the selected frequency ω_i, the execution

time t_{ω_i} of a task v_i varies according to the following formula

$$t_{\omega_i} = \tau_i * \frac{\omega_{max}}{\omega_i} \tag{1}$$

where τ_i is the execution time measured at highest frequency ω_{max}. Then, the energy consumption of a task v_i executed at voltage V_i and frequency ω_i is given by

$$e_i = K * \omega_i * V_i^2 * t_{\omega_i} \tag{2}$$

where K is a constant incorporating the effective capacitance and the activity factor, and t_{ω_i} is the execution time of v_i at frequency ω_i. Finally, the total energy consumption is

$$E = \sum_{v_i \in G_{app}} e_i \tag{3}$$

It is worth mentioning that this work does not take into account communication energy.

C. Susceptibility to Transient Faults

The occurrence of transient faults is commonly modeled by means of a Poisson distribution with occurrence rate λ [6], [7], [12]. As investigated in the past (refer to [6]), the occurrence rate highly depends on the frequency (and hence the voltage) of operation, according to the following equation:

$$\lambda(\omega_i) = \lambda_0 10^{\frac{d(1-\omega_i)}{1-\omega_{min}}} \tag{4}$$

where λ_0 is the soft-error rate at the maximum frequency level ω_{max}, d is architecture specific constant stating the susceptibility to transient faults due to the frequency scaling, and ω_i is the frequency level the task is executed (in the above formula frequency levels are normalized so that $\omega_{max} = 1$). Thus, the reliability of the task v_i executed at frequency ω_i under the influence of transient faults is given by:

$$R_{v_i}(\omega_i) = e^{-\lambda(\omega)\frac{\tau_i}{\omega_i}} \tag{5}$$

D. System Aging

In this paper, wear-out failures related to electromigration (EM) are considered. The phenomenon refers to the movement of metal atoms from the interconnect wires and vias due to the flow of current through it. Indeed, any other effect (time-dependent dielectric breakdown, stress migration, and thermal cycling) can be easily incorporated in the model according to the *sum-of-failure-rates* (SOFR) model [15].

The fault density due to EM is characterized by a Weibull distribution with scale parameter given by the following equation (refer to [15]).

$$\alpha(T_i) = \frac{A_0 (J - J_{crit})^{-n} e^{\frac{E_a}{KT_i}}}{\Gamma\left(1 + \frac{1}{\beta}\right)} \tag{6}$$

where A_0 and n are material-related constant, $J(J_{crit})$ is the (critical) current density, E_a is the activation energy, K is the Boltzman's constant and T_i is the current temperature of the core. Thus, the lifetime reliability of a core is given by:

$$R_{lifetime}(t) = e^{-(At)^\beta} \tag{7}$$

where the time t is measured in number of executed iteration of the application, β is the slope parameter of the Weibull distribution and A is the aging rate of the core per iteration of the application and is given by (refer to [4], [5], [16]):

$$A = \frac{\sum_i \frac{\Delta t_i}{\alpha(T_i)}}{t_p} \tag{8}$$

where t_p is the period of the application graph, Δt_i the time intervals within period t_p and T_i are the related core temperatures. It is worth noting that the temperature profile during the execution period is one of the important components of the aging rate of a core, and in turn depends on its voltage and frequency of operation; a specific characterization need to be performed on the architecture under analysis [4], [16], [17].

Thus, the lifetime of a core can be estimated in terms of the mean time to failure (MTTF), determined by the area under the reliability curve defined in Equation 7. Mathematically, the MTTF of core n_j with reliability $R_j(t)$ is given by

$$MTTF_j = \int_0^\infty R_{lifetime\_j}(t)dt = \int_0^\infty e^{-(A_j t)^\beta} dt \tag{9}$$

The MTTF for a multi-core system with $|\mathcal{V}_{arc}|$ cores is determined by the minimum of the MTTFs of the constituent cores (similar to that used in [4], [5]). Throughout the rest of this paper, the MTTF of an MPSoC platform refers to the minimum of the MTTFs of the different cores of the system.

$$MTTF = \min_j \{MTTF_j\} \tag{10}$$

E. Fault tolerance techniques

Redundancy techniques are commonly used for improving the reliability of the system due to transient faults. These techniques mainly act at application level and assume specific support mechanisms to be implemented on the underlying platform; generally they require the architecture to be provided with fault detection mechanisms. Some examples of the commonly used ones are checkpointing, re-executions, or replications applied at task level [9], [10].

In this work, *selective task replication* has been adopted (as in [13]). The technique is applied individually on each task of the application in input $\mathcal{G}_{app}$, and involves replicating the node, and the related input/output dependencies, for a specific number of time k; no voter or checker is required to compare the redundant results since the architectural fault detection mechanisms will signal faulty results. k can be equal to 0, that means that no replica is introduced for the current task, and its upper bound is specified by the designer. The result of the hardening process is a new task graph $\mathcal{G}_{repl\_app}$, that will be considered for the subsequent synthesis activities in place of the original one.

The goal of the technique is to improve the reliability of the system due to transient faults, and, at the same time, to limit the negative effect the execution of a higher number of tasks has on the architecture's aging. In particular, given a task v_i executed at frequency ω_i and with its k replicas v_i^j each

one working at a given frequency ω_i^j, the reliability due to transient faults can be computed as

$$R_{v_i}^{repl} = 1 - (1 - R_{v_i}(\omega_i)) \cdot \prod_{j=1}^{k} \left(1 - R_{v_i^j}\left(\omega_i^j\right)\right) \quad (11)$$

Therefore, the overall reliability of the system due to transient faults, also called *performability* [6], is computed as:

$$R_{system} = \prod_{v_i \in \mathcal{G}_{repl\_app}} R_i^{repl} \quad (12)$$

Finally, the impact of the introduced replicas on the lifetime is quantified by the previous formulas according to the mapping and the scheduling choices of the hardened application on the given architecture.

IV. Problem Formulation

Given: An application represented as a directed acyclic graph $\mathcal{G}_{app}$, an architecture graph $\mathcal{G}_{arc}$ composed of $|\mathcal{V}_{arc}|$ cores, a given energy budget E_{max} and a deadline represented as the steady state period P_{max}.

Objective: The objective is to determine the following

1. Tasks of $\mathcal{G}_{app}$ that needs to be replicated, thus generating a new task graph $\mathcal{G}_{repl\_app}$.
2. Voltage and frequency of each task of $\mathcal{G}_{repl\_app}$.
3. Mapping and scheduling of $\mathcal{G}_{repl\_app}$ on $\mathcal{G}_{arc}$.

Such That:

1. The lifetime, measured in terms of MTTF, is maximized.
2. The reliability due to transient faults, that is the performability, is maximized.
3. Tasks' precedences are satisfied in the application execution.
4. The energy consumption E of the application is smaller than the energy budget, i.e. $E \leq E_{max}$.
5. The execution period P is within the specified deadline, i.e., $P \leq P_{max}$.

V. The Design Space Exploration Engine

An engine based on a multi-objective genetic algorithm has been designed to automate the described DSE process by evaluating at the same time both the two reliability aspects. In particular, the Non-dominated Sorting Genetic Algorithm (NSGA-II [18]) has been adopted, due to its capability to perform an efficient multi-objective optimization; indeed it features an advanced ranking and elitism mechanisms that allow to preserve diversity into the pool of candidates solutions (i.e., the population) considered during the design space exploration, thus supporting the exploration of several directions minimizing the risk of getting stuck in local minima.

Each candidate solution is encoded in a *chromosome* consisting of a fixed-length vector of genes, each one describing the selected configuration for a task of the application in input (as shown in Figure 3). The *gene* is an aggregate structure containing two fields for representing the mapping and the frequency/voltage level of the nominal task, and a vector of pairs with a variable size, describing the replicas of the task. More precisely, the length of that vector specifies the number of replicas to be instantiated, and each contained pair represents the mapping and the frequency/voltage level of a specific replica.

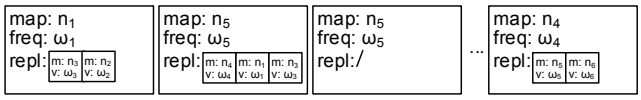

Figure 3. Chromosome structure.

The solution is evaluated by means of a *fitness function* that builds the configuration described in the chromosome starting from the specifications of the application and the architecture given in input, and performs a simulation of its execution, on the basis of the models presented in Section III, to evaluate the considered metrics. In particular, the application is scheduled with a list-based scheduling algorithm to compute the various metrics, i.e., execution period, energy consumption, lifetime and performability. The genetic algorithm compares solutions by means of the Pareto dominance criterion based on two *objectives*: the maximization of the system lifetime (Equation 10) and the maximization of the performability (Equation 12). At the same time, in order to be feasible, the solution must fulfill two constraints on the execution period P and on the computation energy E that must be within a specified deadline P_{max} and energy budget E_{max}. To differentiate unfeasible solutions during the selection process, NSGA-II supports the use of a violating index to rank unfeasible solutions in place of the above-mentioned objectives. The following formula is used for this purpose:

$$V = \frac{P}{P_{max}} \cdot \frac{E}{E_{max}} \quad (13)$$

To generate new solutions the following mutation and crossover operators have been defined. The *mutation* operator, applied with a probability P_M, has been implemented to change the various fields composing each gene. In particular, it may change

- the mapping and the frequency/voltage level of a nominal task,
- the number of replicas to be inserted for a task, and
- the mapping and the frequency/voltage level of each replica of a task.

When decreasing the number of replicas, the related mapping and frequency/voltage level data to be deleted are randomly selected in the vector, while when increasing that number, new mapping and voltage level values are randomly generated. Finally, each solution may be fixed to guarantee that related replicas are mapped on different processing units. The *crossover* operator, applied with a probability P_C, has been implemented as a single point crossover, that is, the two parent chromosomes are cut in the same place and the substrings are exchanged to generate two children chromosomes.

The defined engine follows the standard NSGA-II workflow. The initial population is generated randomly, and, then, the evolution is carried out for a specified number of generations

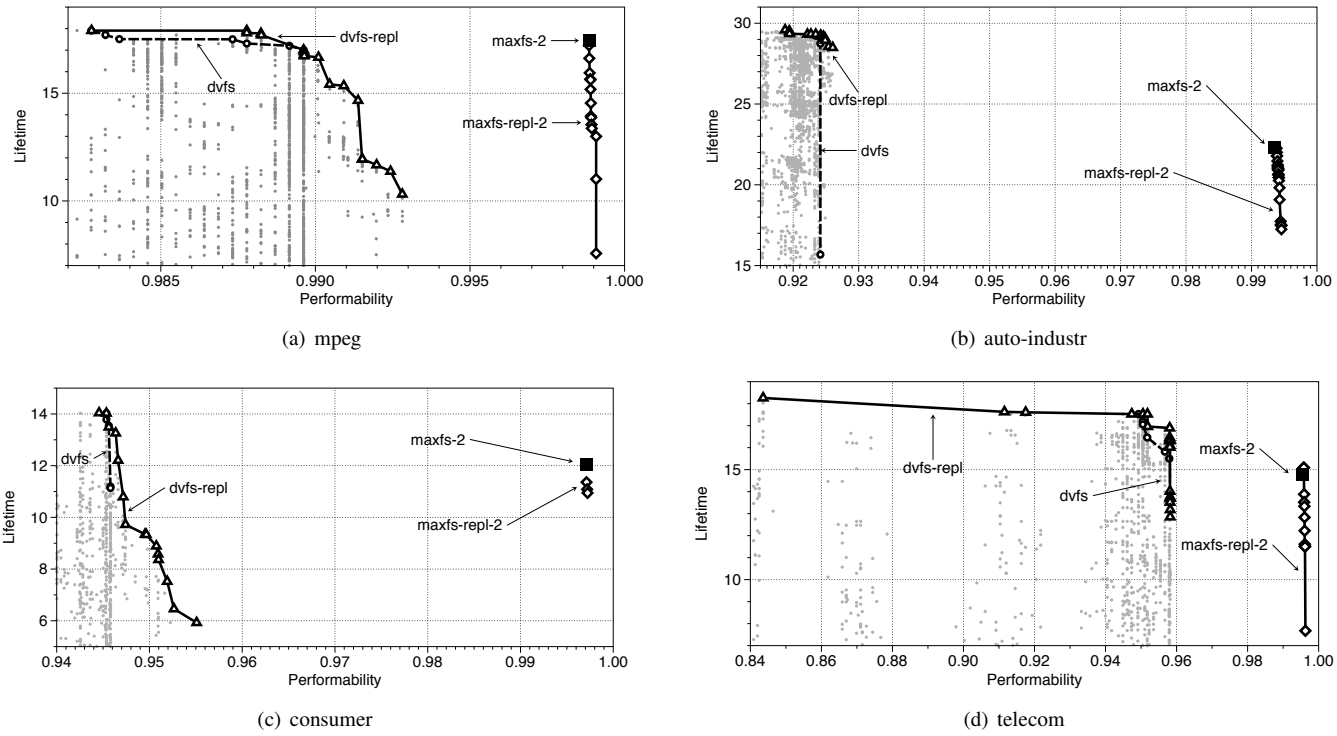

Figure 4. Esperimental results on 4 real-life applications.

by applying the described operators. Due to the large number of variables in the chromosome structure, the design space may grow dramatically with the number of tasks in the application and nodes in the architecture; thus, to improve the efficiency of the exploration process, we guarantee that the initial population contains a certain percentage of feasible solutions. Moreover, at the beginning the replication is disabled to perform an optimization of the frequency/voltage levels only. Later, after a given number of generations, replication is enabled. At the end of the automatic exploration, the engine returns a reduced set of best Pareto-dominant solutions, that can be manually analyzed by the designer to identify the one with the desired lifetime/soft-error susceptibility trade-off.

VI. Experimental Results

The proposed automation engine has been implemented in a C++ prototype integrated with a SystemC functional simulator of the described system models, and it has been used for the experimental sessions considering both synthetic and real-life applications to evaluate the proposed methodology.

Here, we report the results obtained on four real-life applications selected from [5] and from the Embedded System Synthesis Benchmark Suite (E3S, [19]). The applications are *mpeg*, *auto-industr*, *consumer-networking* and *telecom*, and they are composed of 15, 16, 20, 17 tasks, respectively, with an execution time at the maximum frequency spanning from $10ns$ to $1000ns$. To execute such applications, we considered an architecture with a 3×2 mesh-topology consisting of processing nodes characterized by the following operating modes (frequency/voltage levels): $900Mhz/1.2V$, $600Mhz/1.1V$, $300Mhz/1.06V$ (characterization from the ARM Cortex reference manual). Furthermore, parameters for computing aging were set as in [5]: current density $J = 1.5 \times 10^6 A/cm^2$, activation energy $E_a = 0.48eV$, slope parameter $\beta = 2$ and $n = 1.1$; the thermal characterization of the architecture has been performed by means of HotSpot [17], by taking into account the self-activity of each core, the operating frequencies and the neighbor cores' temperature. For the transient faults, the following parameters where used (as in [20]): $\lambda_0 = 10^{-6}$ and $d = 3$. Finally, in this proposed experimental session we set the maximum number of replicas equal to 1; we leave as a future work the investigation of the results and of the exploration efficiency with a higher number of replicas.

For each application, the deadline is set by increasing by a 20% the best possible period on the given architecture (it was computed by means of a preliminary performance-driven DSE without considering any constraint). Similarly, the best energy value was computed by means of another specific DSE by taking into account the above-mentioned deadline, and the best MTTF possible by setting all the tasks to the lowest frequency and running another DSE without any deadline constraint (otherwise no feasible solution would be found); thus, the energy budget E_{max} has been set by increasing by a 50% the best energy value; the best MTTF was considered as reference value in the experimental campaigns (46, 62, 36 and 75 years respectively).

To evaluate the design space exploration capability of the proposed methodology, experiments were conducted with the following engine configurations: (i) the full-fledged en-

gine (labelled *dvfs-repl*), (ii) with replication disabled (*dvfs*), (iii) with replication disabled and using only the maximum voltage/frequency levels (*maxfs*), and (iv) with replication and using the maximum voltage/frequency levels (*maxfs-repl*).

Figure 4 presents the results for the four considered applications. In particular, each graph reports the two Pareto fronts for *dvfs* and *dvfs-repl*, and the solutions that have been explored (actually, the subset closest to the Pareto fronts for sake of space). Moreover, no feasible solution has been found with *maxfs* and *maxfs-repl*, because when working at maximum frequency, the system is not able to meet the given energy budget. To obtain feasible solutions with these last two approaches, the constraint on the energy budget was relaxed to an additional 100% with respect to the initial best energy value; such Pareto curves are reported in the graphs as *maxfs-2* (only a single point with the best MTTF value possible) and *maxfs-repl-2*. When considering the *dvfs* and *dvfs-repl* fronts, it is possible to notice that the latter almost dominates the former, and moreover, it extends to the rightmost part of the graph. This shows that *dvfs-repl* is at the same time able i) to generate the solutions of *dvfs* without using replicas, and ii) to exploit selective replication to increase the performability of the solutions, by leveraging lifetime, thus obtaining a larger set of candidate solutions.

When analysing the *mpeg* application, the maximum MTTF value obtained has half the lifetime with respect to the above-mentioned reference values. Indeed, to fulfill the performance deadline, voltage/frequency levels need to be increased and therefore the processing units reaches higher temperatures, thus causing a higher wear-out than the ideal (not feasible) configuration. Moreover, the rightmost points on the *dvfs-repl* front experience a decreasing MTTF trend (3 times shorter), since the architecture stress is exacerbated by the introduction of replicas and the additional voltage/frequency increasing. At the same time, these choices offer the possibility to considerably improve the performability of the system (from 0.984 to 0.992), thus getting closer to *maxfs-2* and *maxfs-repl-2* fronts (that may be considered as the ideal reference values). In conclusion, the designer is provided with a set of interesting solutions; the most suitable one can be chosen according to the requirements. Similar considerations can be drawn for the other applications.

VII. Conclusions

This paper proposed an approach to map a given set of applications on a MPSoC platform to jointly optimize the aging of cores and soft-error susceptibility. The proposed methodology selects multiple mapping solutions providing designers the flexibility to select a mapping according to his/her needs in terms of lifetime/number of errors experienced. Experiments conducted with real-life application clearly demonstrate the advantage of the proposed model. In future, heterogeneous architectures and alternative redundancy techniques will be explored, and the performance of the automation engine will be further evaluated; nevertheless, the approach will be enhanced to consider graceful degradation, by defining remapping strategies to be adopted after the occurrence of permanent failures.

Acknowledgment

This work was supported by Singapore Ministry of Education Academic Research Fund Tier 1 with grant number R-263-000-655-133.

References

[1] M. Schmitz, B. Al-Hashimi, and P. Eles, "Energy-efficient mapping and scheduling for dvs enabled distributed embedded systems," in *Proc. Conf. Design, Automation and Test in Europe*, 2002, pp. 514–521.

[2] A. K. Singh, A. Das, and A. Kumar, "Energy optimization by exploiting execution slacks in streaming applications on multiprocessor systems," in *Proc. Design Automation Conference*, 2013, pp. 115:1–115:7.

[3] S. Borkar, T. Karnik, and V. De, "Design and reliability challenges in nanometer technologies," in *Proc. Design Aut. Conf.*, 2004, pp. 75–75.

[4] L. Huang, F. Yuan, and Q. Xu, "On Task Allocation and Scheduling for Lifetime Extension of Platform-Based MPSoC Designs," *IEEE Trans. Parallel and Distributed Systems*, vol. 22, no. 12, pp. 2088–2099, 2011.

[5] A. Das, A. Kumar, and A. Veeravalli, "Reliability-Driven Task Mapping for Lifetime Extension of Networks-on-Chip Based Multiprocessor Systems," in *Proc. Conf. Design, Automation and Test in Europe*, 2013.

[6] D. Zhu, R. Melhem, and D. Mosse, "The Effects of Energy Management on Reliability in Real-time Embedded Systems," in *Proc. Conf. on Computer-aided Design*, 2004, pp. 35–40.

[7] B. Zhao, H. Aydin, and D. Zhu, "Generalized reliability-oriented energy management for real-time embedded applications," in *Proc. Design Automation Conference*, 2011, pp. 381–386.

[8] Y. Zhang and K. Chakrabarty, "Energy-aware adaptive checkpointing in embedded real-time systems," in *Proc. Conf. Design, Automation and Test in Europe*, 2003, pp. 10 918–10 925.

[9] V. Izosimov, I. Polian, P. Pop, P. Eles, and Z. Peng, "Analysis and optimization of fault-tolerant embedded systems with hardened processors," in *Proc. Conf. Design, Autom. and Test in Europe*, 2009, pp. 682–687.

[10] C. Bolchini and A. Miele, "Reliability-driven system-level synthesis of embedded systems," in *Proc. Symp. Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, 2010, pp. 35–43.

[11] J. Huang, J. Blech, A. Raabe, C. Buckl, and A. Knoll, "Analysis and optimization of fault-tolerant task scheduling on multiprocessor embedded systems," in *Proc. Conf. Hardware/Software Codesign and System Synthesis*, 2011, pp. 247–256.

[12] A. Das, A. Kumar, and B. Veeravalli, "Aging-aware hardware-software task partitioning for reliable reconfigurable multiprocessor systems," in *Proc. Conf. Compilers, Architectures and Synthesis for Embedded Systems*, 2013.

[13] J. Gan, F. Gruian, P. Pop, and J. Madsen, "Energy/reliability trade-offs in fault-tolerant event-triggered distributed embedded systems," in *Proc. Asia South Pacific Design Automation Conf.*, 2011, pp. 731–736.

[14] N. Iqbal, M. Siddique, and J. Henkel, "SEAL: Soft error aware low power scheduling by Monte Carlo state space under the influence of stochastic spatial and temporal dependencies," in *Proc. Design Automation Conference*, 2011, pp. 134–139.

[15] JEDEC Solid State Technology Association and others, "Failure mechanisms and models for semiconductor devices," *JEDEC Publication JEP122-B*, 2003.

[16] A. Das, A. Kumar, and B. Veeravalli, "Temperature aware energy-reliability trade-offs for mapping of throughput-constrained applications on multimedia mpsocs," in *Proc. Conf. Design, Automation and Test in Europe*, 2014.

[17] K. Skadron, M. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: Modeling and implementation," *ACM Trans. Archit. Code Optim.*, pp. 94–125, 2004.

[18] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[19] Embedded System Synthesis Benchmark Suite web site, "http://ziyang.eecs.umich.edu/~dickrp/e3s/," accessed: 2013-09-19.

[20] B. Zhao, H. Aydin, and D. Zhu, "Energy Management Under General Task-Level Reliability Constraints," in *Proc. Real Time and Embedded Technology and Applications Symp.*, 2012, pp. 285–294.