Reliable Multiprocessor System-On-Chip Synthesis

Changyun Zhu†   Zhenyu (Peter) Gu‡   Robert P. Dick‡   Li Shang†

† ECE Department
Queen's University
Kingston, ON K7L 3N6, Canada
changyun.zhu@ece.queensu.ca
li.shang@queensu.ca

‡ EECS Department
Northwestern University
Evanston, IL 60208, U.S.A.
zgu646@eecs.northwestern.edu
dickrp@northwestern.edu

ABSTRACT

This article presents a multiprocessor system-on-chip synthesis (MPSoC) algorithm that optimizes system mean time to failure. Given a set of directed acyclic periodic graphs of communicating tasks, the proposed algorithm determines a processor core allocation, level of system-level and processor-level structural redundancy, assignment of tasks to processors, floorplan, and schedule in order to minimize system failure rate and area while meeting functionality and timing constraints. Changes to the thermal profile resulting from changes in allocation, assignment, scheduling, and floorplan are modeled and optimized during synthesis, as is the impact of thermal profile on temperature-dependent failure mechanisms. The proposed techniques have the potential to substantially increase MPSoC system mean time to failure compared to area-optimized solutions. If power densities are high and the dominant lifetime failure mechanisms are strongly dependent on temperature, our results indicate that thermal and structural redundancy optimization during synthesis have the potential to greatly increase MPSoC lifetime with low area cost.

Categories and Subject Descriptors: C.4 [Performance of Systems]: Reliability, availability, and serviceability; Fault tolerance. C.5.4 [Computer System Implementation]: VLSI Systems.
General Terms: Reliability, Algorithms, Design, Performance

1. INTRODUCTION

A single integrated circuit can now contain over 500 million transistors. The International Technology Roadmap for Semiconductors (ITRS) [1] projects billion-transistor integrated circuits by 2007. It has been necessary to move to MPSoCs to control design complexity and power consumption.

Increasing power density due to continued scaling of CMOS process technology accelerates temperature-dependent and current-dependent failure mechanisms such as electromigration. Lifetime reliability is becoming an important quality metric in high-performance MPSoCs. Optimizing lifetime reliability requires careful planning during MPSoC design and synthesis. This problem cannot be well solved at any single level of the design process. Reliability characterization requires MPSoC thermal profiles, which in turn requires physical information, including an MPSoC floorplan, power profile, and chip-package thermal model. Reliability-aware MPSoC design requires a unified architectural-level and physical-level design flow.

1.1   Past Work and Contributions

Our work draws from research in the areas of integrated circuit reliability modeling and optimization [2,3], system synthesis [4–7], physical design, and thermal analysis [8, 9]. Coskun et al. [3] and Srinivasan et al. [2] provide architectural reliability models and run-time optimization techniques for MPSoCs and microprocessors, respectively. Eles et al. contrast optimization algorithms for use in hardware–software partitioning [4]. Henkel and Ernst propose flexible task discretization during hardware–software partitioning [5]. Xie et al. propose a technique to duplicate tasks on idle processors during embedded system synthesis to tolerate transient faults [6]. Lee and Ha propose an allocation, assignment, and scheduling algorithm for real-time MPSoCs [7]. Ogras et al. propose a branch-and-bound algorithm for NoC synthesis [10]. Glaß et al. propose an evolutionary algorithm that binds tasks to resources with the goal of improving mean time to failure (MTTF) [11]. They consider fault processes with exponential or Weibull distributions; their fault model supports permanent faults. Our system and fault model differs primarily by considering the influence of faults on subsequent fault rates due to the impact of run-time rebinding on temperature profile.

Our work addresses synthesis of MPSoCs capable of reliable operation in the presence of permanent faults. The proposed algorithm generates MPSoC architectures that satisfy the functionality and performance constraints of a specification while simultaneously optimizing die area and MTTF. The problem specification consists of graphs composed of data-dependent, multirate, periodic tasks as well as a database of processor cores. Each processor core executes different tasks with different execution times and power consumptions. This work makes the following main contributions.

1. We have developed and implemented an MPSoC synthesis flow that conducts architectural synthesis, floorplanning, on-chip network synthesis, chip-package thermal analysis, and reliability analysis. Optimization algorithms within this flow exploit redundancy and temperature-aware design planning to produce reliable, compact MPSoC designs.

2. We propose a two-phase reliability optimization flow that builds on a stochastic functionality, performance, and area optimization algorithm and an iterative reliability enhancement algorithm that explores the trade-off between MPSoC reliability and area. This algorithm improves MPSoC system MTTF by an average of 85% with less than 5% area cost and by an average of 436% with less than 25% area cost, compared to area-optimized solutions.

To the best of our knowledge, this is the first work to propose and implement a method of predicting and optimizing the impact of design changes during synthesis on temperature-dependent MPSoC failure processes.

1.2   System MTTF Definition and Example

We define *system MTTF* to be the expected amount of time an MPSoC will operate, possibly in the presence of component faults, before its performance drops below some designer-specified constraint or it is no longer able to meet it functionality requirements. Using system MTTF to characterize reliability has the advantage of taking into account performance; this is important for consumer electronics and most other MPSoC applications.

Figure 1: Reliable MPSoC synthesis example.

To concurrently optimize the system MTTF and area of an MPSoC, it is necessary to exploit both hardware redundancy and temperature profile. *Processor-level redundancy* is achieved by adding processors to the MPSoC architecture. *Component-level redundancy* is achieved by adding appropriate control mechanisms and redundant hardware such as additional arithmetic logic units (ALUs) or cache banks to individual processors [2]. We will illustrate each method of improving system MTTF using an example. Figure 1 shows two synthesized solutions for a telecommunication application based processor performance data from the Embedded Microprocessor Benchmark Consortium [12]. Each solution contains three embedded processors connected by an on-chip router. The temperature of each on-chip component is indicated by its brightness: brighter components are hotter. The embedded processor, an AMD K6-2E+, used in Solution I, is replaced with an IBM PowerPC 405GP-RE in Solution II. 405GP-RE is a low power, redundant version of the 405GP; the floating/fixed point units and register files are duplicated. The system MTTFs of Solution I and Solution II are 0.7 year and 1.5 years; these changes doubled MTTF. Further reliability enhancements can be used to increase MTTF to 7 years at small area cost.

This example illustrates the potential improvement to system MTTF due to temperature reduction and resource redundancy. MPSoC reliability strongly depends on temperature. In Solution I, the K5-2E+ has a peak temperature of 59.9 °C. In Solution II, replacing the K5-2E+ with the 405GP-RE reduces the peak temperature by 5.1 °C, thereby decreasing the run-time fault rate. Second, increasing system redundancy improves fault-tolerance. Compared to the K5-2E+, the 405GP-RE can tolerate more run-time faults. This results in an improvement to system MTTF.

2. TASR: TEMPERATURE-AWARE SYNTHESIS OF RELIABLE MPSOCS

In this section, we describe TASR, the proposed reliable application-specific MPSoC synthesis infrastructure.

2.1 TASR Infrastructure

Determining and optimizing MPSoC system MTTF requires substantial infrastructure. Figure 2 illustrates some of the main steps and components in the proposed synthesis flow. Computing system MTTF requires knowledge of component MTTFs and run-time performance constraints. Computing component MT-TFs requires knowledge of MPSoC thermal profile and architecture. Computing MPSoC thermal profile during synthesis requires a floorplan, task assignment dependent power modeling, and a thermal analysis algorithm. Finally, determining, and optimizing MPSoC architecture requires a system-level synthesis infrastructure that allocates processor cores, assigns tasks to processors, rapidly generates floorplans, assigns communication events to network links, and schedules operations and communication events.

TASR is composed of algorithms from three domains: system-level synthesis, physical synthesis, and solution analysis. The system-level design contains a single-objective stochastic optimization algorithm that minimizes MPSoC area subject to functionality and performance requirements, and an iterative reliability enhancement algorithm that uses knowledge of redundancy and thermal profile to improve system MTTF at a small cost in MPSoC area. Physical-level synthesis consists of a slicing floorplanning algorithm and an on-chip network synthesis algorithm. In addi-

tion, TASR contains a novel statistical lifetime reliability model, and also performance, power, and thermal models to guide MPSoC reliability optimization.

Given

1. Functionality and timing requirements consisting of a directed acyclic graph of periodic graphs of communicating heterogeneous tasks, each of which may have a different deadline;

2. Databases indicating the properties of the available heterogeneous processor cores and on-chip network resources when used with the tasks in the functionality requirements specification, e.g., task execution times and power consumptions on each processor and processor areas; and

3. Temperature-dependent reliability models for the processors and functional units within them

TASR uses a two-stage optimization flow to determine

1. An allocation of processor cores that are selected based on their performance and reliability characteristics;

2. An assignment of tasks to processor cores that takes task impact on temperature and therefore reliability into account;

3. A schedule of all the tasks and communication events in the system; and

4. A floorplan for the MPSoC.

The solutions are optimized for reliability (maximized MTTF) and area. Each solution is associated with numerous alternative task assignments and schedules to permit continued operation in the event of processor core failure. If a processor fails, the resulting change in task assignment and schedule required to maintain functional correctness and meet timing requirements is pre-planned.

2.2 Two-Phase Synthesis Flow

This section explains the two-phase synthesis process used within TASR. The first phase uses a parallel recombinative simulated annealing (PRSA) algorithm, i.e., an advanced form of genetic algorithm, to search for low-area MPSoC architectures that meet functionality and timing requirements without violating area constraints. Previous studies [13] have demonstrated that the use of PRSA allocation and assignment together with adaptive list scheduling permits optimal solutions to problems for which optimal solutions are known [14]. For problem instances with previously published results, the PRSA approach rapidly produces solutions of equal or better quality [15, 16]. Adaptive list scheduling makes multiple scheduling attempts with different prioritization metrics in order to meet timing and functionality constraints.

The MPSoC lifetime reliability optimization problem can potentially be solved using a PRSA synthesis flow by including system MTTF with the other optimization objectives. However, the addition of reliability optimization to functional, timing, and area optimization greatly increases problem complexity. Moreover, the time cost of determining the reliability impact of a design change is much higher than that of determining the area and performance impact. It becomes necessary to conduct thermal and reliability analysis and to determine multiple task assignments and schedules for each MPSoC in order to support runtime adaptation to processor core failure. Therefore, we propose starting from an area-optimized solution meeting functionality and timing constraints and using a reliability enhancement algorithm to explore the area–reliability tradeoff curve.

Lifetime reliability is inversely related to chip temperature. By increasing chip area, power density and chip temperature decrease, thereby increasing chip reliability. Structural redundancy, which permits continued processor or MPSoC operation after component failure and generally increases area, can also improve reliability.

2.3 MPSoC Reliability Analysis

MPSoC lifetime reliability is a function of various design-time and run-time parameters, including failure mechanism properties, resource redundancy, and chip thermal profile. In TASR, MPSoC lifetime reliability optimization is guided by a statistical lifetime reliability model that takes multiple failure mechanisms into consideration [17]. This section briefly summarizes the main attributes
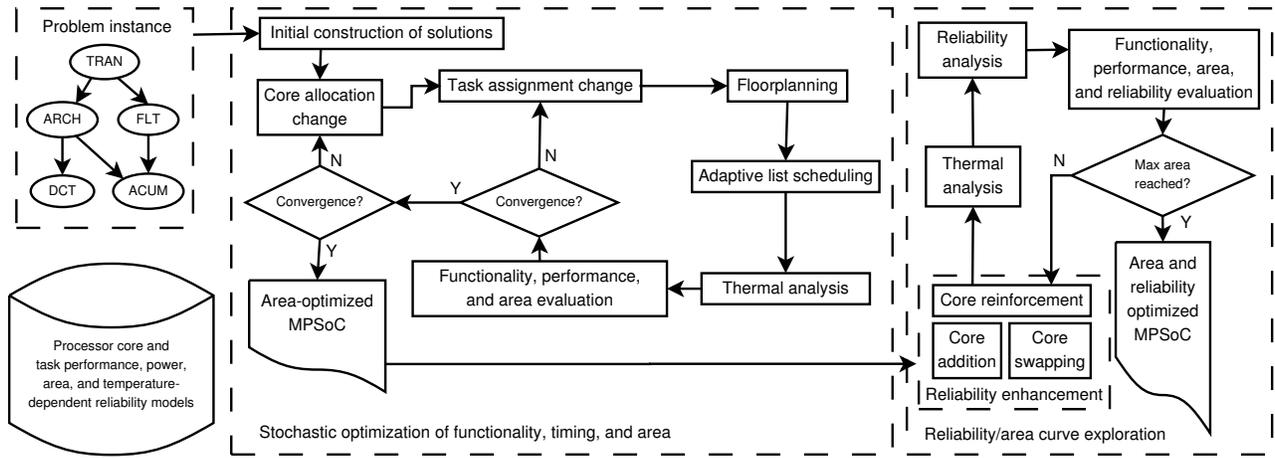
Figure 2: TASR flow for the temperature-aware synthesis of reliable MPSoCs.

of this model.

1. Our failure model considers electromigration, thermal cycling, time-dependent dielectric breakdown, and stress migration [18]. Most of these failure processes have MTTF expressions of the form

$$MTTF = K_1 e^{\frac{K_2}{T}} \qquad (1)$$

for temperature T and temperature-independent constants K_1 and K_2 [18]. For instance, the MTTF due to electromigration follows [19]:

$$MTTF_{EM} = \frac{A_{EM}}{J^n} e^{\frac{E_{a_{EM}}}{\kappa T}} \qquad (2)$$

where A_{EM} is a constant determined by the physical characteristics of the metal interconnect, J is the current density, $E_{a_{EM}}$ is the activation energy of electromigration, n is an empirically-determined constant, κ is Boltzmann's constant, and T is the temperature.

As indicated by Item 1, MTTFs depend strongly on temperature. Therefore, an accurate on-line thermal model is required to predict the impact of design changes during synthesis on temperature profile and therefore reliability.

2. A MIN–MAX model is used to characterize structural redundancy. Assume a processor core contains M types of resources. For each type of resource $S_i, i \in \{1, \cdots, M\}$, $F_{S_i}(t)$ is its cumulative failure probability. Thus, the MTTF of this processor core can be estimated using the following equation.

$$MTTF_p = \min_{i=1}^{M} \left(\int_0^1 t \, dF_{S_i}(t) \right) \qquad (3)$$

3. Our model considers wear. Many important microprocessor fault processes accelerate as a result of wear; this complicates reliability modeling and analysis. It is common to model microprocessor faults as Poisson processes, which yield cumulative fault probabilities with exponential distributions. Exponential distributions are mathematically convenient because they permit the rates of different fault processes operating on different components to be added in order to determine the failure rate of the entire microprocessor. However, they do not model wear, which is generally required for accurate reliability modeling [20].

The lognormal distribution is more appropriate for prominent microprocessor fault processes because it models the increase in failure rate with increasing time and wear [2]. Our model uses a lognormal distribution for cumulative fault probability as a function of time. Combining the contributions of different components to system-level failure rate is complicated by the use of lognormal processes. There is no straightforward method of deriving a closed-form expression for the failure rate of a microprocessor composed of numerous components using lognormal fault models. We use curve fitting with table lookups on precomputed MTTF values based on varying voltages and temperatures to accelerate the process. However, reliability estimation remains one of the most time consuming stages of the synthesis.

4. When a component fails, our method of determining whether the MPSoC has failed is sophisticated but expensive. We generate a new task assignment and schedule without using the failed component to determine whether MPSoC can still meet its functionality and timing requirements. This allows the use of task migration among heterogeneous components to improve system MTTF.

2.4 Reliability Optimization of MPSoCs

Figure 2 illustrates the proposed reliability analysis and optimization flow. In TASR, reliability optimization starts by evaluating the system MTTF of area optimized solutions (using Algorithm 1), Such solutions tend to have high power density, high temperature, low resource redundancy and, therefore, low system MTTF. An iterative reliability enhancement algorithm is invoked if these solutions do not provide the required system MTTF. During each iteration, Algorithm 2 optimizes MTTF by improving processor core and component redundancy and/or optimizing chip thermal profile by introducing new processors. System-level (task assignment and scheduling) and physical-level (floorplanning and network synthesis) algorithms are then invoked to produce valid MPSoC solutions. Through performance, power, thermal, and reliability analyses, the system MTTFs of new solutions are estimated and evaluated. The iterative optimization flow continues until the targeted system MTTF is achieved.

Algorithm 1 estimates system MTTF based on statistical models of MPSoC run-time failure processes. Starting from time $t = 0$, it determines the minimal MTTF among all the processor cores (line 4). Each fault may result in partial or complete processor core failure. In either case, task migration is used to optimize system performance. The task migration routine moves tasks from the faulty or partially-faulty processor to other processors (line 6). After task migration, if the MPSoC still meets its performance requirements, the algorithm considers the next processor core with minimal MTTF. Task migration results in run-time changes in chip power consumption and temperature profiles, thereby changing the lifetime reliability of each processor core. To accurately predict subsequent processor MTTFs, power and thermal analysis are conducted (line 8). This process continues until the MPSoC fails to meet its performance or functionality requirements. The system MTTF of the MPSoC solution is then reported (line 11). Following all paths in the processor failure lattice during synthesis would have prohibitive computational complexity. Partially exploring paths associated with lower probabilities of failure might improve the accuracy of MTTF estimation. This is the subject of ongoing work.

At run-time, on-line fault detection algorithms should determine when an execution unit has failed. A proper treatment of on-line fault detection is beyond the scope of this work but can be found in the literature [21]. Upon fault detection, the pre-planned task

Algorithm 1 System MTTF analysis of an MPSoC solution
___
1: Given an MPSoC solution, set $MTTF_{MPSoC} \leftarrow 0$
2: **while** system schedule is valid **do**
3:     $MPSoC_{Func}$ are the functioning processors in the MPSOC
4:     Fault interval $e_i \leftarrow \min_{p \in MPSoC_{Func}}(MTTF_p)$
5:     $MTTF_{MPSoC} \leftarrow MTTF_{MPSoC} + e_i$
6:     Task migration, scheduling
7:     **if** system scheduling is valid **then**
8:         Power analysis, thermal analysis, compute processor temperatures
9:     **else**
10:         Return $MTTF_{MPSoC}$
11:     **end if**
12: **end while**
___

Algorithm 2 Reliability-aware optimization algorithm
___
1: **while** $MTTF_{MPSoC} < MTTF_{target}$ **do**
2:     $\forall_{pe \in MPSoC}$ compute $MTTF_{pe}$
3:     Find vulnerable point: P_{vul} is the processor with minimal MTTF
4:     Optimization moves (processor reinforcement, processor swapping, processor addition)
5:     Apply the best move based on Equation 4
6:     System-level synthesis: Task assignment and Scheduling
7:     Physical-level synthesis: Floorplanning and network synthesis
8:     Performance, power, thermal, reliability analysis
9:     **if** system MTTF does not improve or system schedule invalid **then**
10:         Revert this change
11:     **end if**
12: **end while**
___

assignment changes associated with the particular fault are made. If it is acceptable to reboot the system in the presence of a fault (a few times in the system lifespan), no further provisions are necessary. If uninterrupted operation is necessary, distributed system checkpointing may be used.

TASR is equipped with an efficient workload migration algorithm to maintain system functionality and meet performance requirements in the presence of partial and complete processor failures. When an MPSoC fails to meet its performance requirements due to run-time faults, tasks migrate to other processors using the following policy. Tasks on faulty processors are first sorted in order of increasing time slack, the difference between the task's latest finish time and earliest finish time. They are then migrated from the processor to other processors in this order until the system performance requirements are met and no tasks are assigned to a totally failed processor. When moving a task from one processor to another, the new processor is selected by Pareto-ranking processors in order of increasing utilization ratio (the proportion of time during which the processor is actively executing tasks) and increasing execution time for the task and processor under consideration. Depending on whether a processor is inoperational or partially-failed, all or some of the tasks assigned to it migrate to other processors.

TASR optimizes the lifetime reliability of MPSoCs by focusing on architectural changes that improve redundancy and thermal profile, while maintaining low area overhead. Algorithm 2 shows the actions taken by TASR to improve the MTTF of an MPSoC architecture. First, the MTTF of each individual processor is estimated (line 2). The processor with the minimal MTTF is identified as the MPSoC's most vulnerable point, P_{vul} (line 3). One of the proposed reliability optimization moves is then applied: processor reinforcement, processor swapping, and processor addition (line 4). *processor reinforcement* introduces component redundancy (see Section 1.2) into the most vulnerable processor. *Processor swapping* replaces the most vulnerable processor with a different, more reliable, processor. *Processor addition* introduces a new processor into the MPSoC, enabling tasks to migrate from the vulnerable processor to other processors. These moves consider multiple candidates processors. TASR uses the relative reliability gain, defined in Equation 4, to select the best candidate move. This equation takes power density reduction, resource redundancy improvement, and area overhead associated with the move into consideration.

$$G_{TASR} = e^{-P_d} \times MTTF_{ref}/A \qquad (4)$$

Note that this value is used only to guide changes. The detailed effect of each tentative change is computed using thermal profile and reliability analysis. MPSoC power profile influences MPSoC temperature profile, which strongly influences reliability. The MTTFs associated with some major fault mechanisms are exponential functions of temperature. Therefore, in Equation 4, TASR uses an exponential term, e^{-P_d}, to characterize the impact of power density reduction on reliability improvement. P_d is the power density reduction resulting from applying a candidate move. In Equation 4, the impact of redundancy is characterized by the second term, $MTTF_{ref}$, the system MTTF improvement resulting from the candidate move. $MTTF_{ref}$ is calculated under the assumption that other design characteristics, e.g., temperature profile and supply

voltage, remain the same. The relative reliability gain introduced by each candidate move is the product of these two terms divided by the area overhead. The move with the highest gain is applied (line 5). After each optimization move, system-level and physical-level synthesis algorithms are invoked to update the MPSoC solution. Cost analysis is then conducted to determine the improvement in system reliability, determine the impact on MPSoC area, and validate the system schedule. This optimization process continues until the target system MTTF is achieved.

Two additional other optimization moves were implemented for the sake of comparison. The first considers only power density, e^{-P_d}, and the second considers only resource redundancy, $MTTF_{ref}$. Performance comparisons among these three heuristics are provided in Section 3.

2.5 Floorplanning, Thermal Analysis, and Network Synthesis

We use a fast constructive area and communication aware floorplanning block placement algorithm based on network partitioning and optimal processor orientation and rotation selection to determine MPSoC power profile as well as communication latency and communication power consumption [13]. A fine-grained MPSoC thermal model is used within a thermal analysis algorithm designed for accuracy and high enough speed for use within the inner loop of synthesis [9]. Finally, we carry out on-chip network synthesis, using network topology to explicitly model communication contention. Although these algorithms are used within TASR, detailed descriptions are omitted due to space constraints.

3. EXPERIMENTAL RESULTS

This section describes the benchmarks used to evaluate TASR and presents the results of evaluation.

3.1 Benchmarks

The proposed reliable MPSoC synthesis algorithm was evaluated using a number of benchmarks from the E3S embedded systems benchmark suite, which is based on EEMBC benchmark data [12]. This suite contains 17 PEs, e.g., the AMD ElanSC520, Analog Devices 21065L, the Motorola MPC555, and the Texas Instruments TMS320C6203. These processors are characterized based on the measured execution times of 47 tasks commonly encountered in embedded applications, power numbers derived from datasheets, and additional information, e.g., processor areas, some of which were necessarily estimated, and prices gathered by emailing and calling vendors. Any processor for which the datasheet reflected results in coarser technologies were linearly scaled to a $0.18\,\mu m$ technology. The task sets follow the organization of the EEMBC benchmarks. There is one task set for each of the five application suites: Automotive/Industrial, Consumer, Networking, Office Automation, and Telecommunications. The Office Automation problem contains only five tasks. Our modified version of Office Automation contains four copies of the original task set. In addition, TGFF [22] was used to generate five random benchmarks, each of which has 30–50 tasks. The graphs have different structures, rang-

Table 1: System MTTF Improvement Under Area Bound

Area bound (%)	MTTF improve. (%)	Area bound (%)	MTTF improve. (%)	Area bound (%)	MTTF improve. (%)
0.0	40.0	15.0	180.0	30.0	457.0
5.0	85.0	20.0	240.0	35.0	468.0
10.0	180.0	25.0	436.0	40.0	470.0

The MTTF improvement under each area bound is computed by selecting the highest-MTTF solution for each benchmark, that honors the area bound, and computing the average of their MTTF improvements.

ing from random connectivity to a series-parallel structure commonly encountered in DSP applications. For the random benchmarks, tasks were randomly assigned task types from the EEMBC benchmarks.

The EEMBC processors do not have component redundancy, i.e., each processor will fail if any of its functional units fails. We introduce a redundant version for each processor by duplicating floating/fixed point units and floating/integer register files. We assume that instruction scheduling units and instruction decode units do not have redundancy [2]; a run-time fault in these units will result in processor failure. On-chip caches have redundancy; a single fault reduces performance but the processor remains operational. We relied on previous work to estimate the cost of component redundancy [2]. Processors with component redundancy suffer a 24% area penalty and, while their additional functional units are still operational, have 25% higher performance and power consumption.

The embedded microprocessors in EEMBC have fairly homogeneous energy–delay products. It is our goal to develop a synthesis algorithm that is effective at improving the reliability of application-specific MPSoCs, which commonly contain heterogeneous processors. Therefore, for each processor, we introduced one corresponding processor operating at a higher voltage and another operating at a lower voltage. A maximum of three voltages need to be provided by off-chip regulators. The alpha power law was used to calculate the impact of voltage scaling on performance. A $0.18\,\mu m$ process, supply voltage of $1.8\,V$, and alpha of 1.3 were used [23]. To model high-performance processors, the supply voltage was scaled to $2.5\,V$, performance increased by 25%, and power consumption increased to $2.4\times$. To model low-power processors, the supply voltage was scaled to $1.28\,V$, performance was decreased by 25%, and power consumption was decreased to $0.38\times$.

3.2 TASR vs. Stochastic Area Optimization

As described in Section 2.1, TASR consists of a two-stage optimization flow. It first uses a stochastic optimization algorithm to minimize MPSoC area under performance constraints. The area-optimized solution is used as a starting point for the proposed reliability enhancements. The TASR lines in Figure 3.2 illustrate the solutions produced by the MTTF optimization technique when run on all the benchmarks. The initial area-optimized solutions appear at the left-most points of the lines. TASR applied the optimization moves described in Section 2.4 until seven subsequent moves did not significantly improve system MTTF. Table 1 shows the average system MTTF improvement over initial area-optimized solutions under different area overhead constraints for all ten benchmarks. These results illustrate three key points about the reliable application-specific MPSoC synthesis problem.

1. The area cost to improve reliability is initially small. In Figure 3.2, area is shown on a logarithmic scale. As shown in Table 1, improving the average system MTTF over all benchmarks by 40%, 85%, and 180% results in maximum area overheads of 0.0%, 5.0%, and 10.0%. MTTF is not directly considered in the first optimization phase. As a result, TASR can sometimes improves MTTF without area overhead because two solutions with the same area can have different MTTFs. Initial solutions are optimized for area and tend to have high power densities, high temperatures, and low resource redundancy: the fault rates are high and single faults may

cause failure. Therefore, the system reliability can be improved at low area cost. TASR introduces processor cores with lower power densities and/or replaces non-redundant cores with redundant ones, thereby optimizing thermal properties and allowing the system to continue operating despite runtime hardware faults.

2. As shown in Table 1, TASR automatically trades off system reliability for area, allowing system designers to choose a desirable solution based on problem-specific design constraints.

3. As system MTTF increases, the area penalty associated with further improving system reliability increases. As shown in Table 1, TASR achieves 436% average system MTTF improvement with a maximum area overhead of 25%. Further improvements to system MTTF become prohibitively expensive. Processor core failure cumulative distribution functions are non-decreasing. For a large enough duration, there is a low probability that any processor will operate without a fault. As a result, at very large MTTFs, adding processors or reinforcing a subset of existing processors with redundant components has little impact on MTTF.

3.3 Evaluation of Optimization Moves

TASR optimizes system reliability by controlling processor temperatures and improving system redundancy. To evaluate the effectiveness of the proposed optimization moves, we compare TASR with two alternative moves described in Section 2.4: power density only (PD-only) and component redundancy only (CR-only) moves. PD-only minimizes power density. CR-only increases resource redundancy. Figure 3.2 shows the results produced by TASR, CR-only, and PD-only optimization moves. TASR almost always produces architectures with both superior area and system MTTF. In some cases, PD-only or CR-only also do well. PD-only does not consider component redundancy. However, introducing redundant processors in order to improve power density still improves system MTTF. CR-only does not consider processor power density. However, redundant processors tend to have lower power densities than non-redundant processors; although their instantaneous spatial power densities are similar to non-redundant processors, they have higher performance, permitting lower temporal power densities. In general, it is necessary to use both structural redundancy and power density to produce high-quality solutions.

3.4 Evaluation of Optimization Flow

As explained in Section 2.2, it appears that a two-phase optimization flow in which a stochastic optimization algorithm is first used to find a promising, low-area, region of the solution space and then an iterative reliability enhancement algorithm is used to trade off area for reliability is superior to a one-phase optimization flow. To determine whether this argument has merit, we compared TASR with a one-phase stochastic optimization algorithm in which functionality, timing, area, and reliability are concurrently optimized. This algorithm, which we call 1PHASE, has the ability to apply all the allocation, assignment, floorplanning, and scheduling changes available to TASR. It optimizes MTTF within its multi-objective cost function. We found that TASR can almost always produce solutions of equal or better quality than 1PHASE. In addition, TASR generally requires less CPU time (an average of 635.9 s per benchmark) than 1PHASE (an average of 2,394 s per benchmark).

4. CONCLUSIONS AND FUTURE WORK

This article has described a synthesis algorithm for reliable application-specific MPSoCs. The dominant failure processes today, and in the near future, have rates exponentially dependent on temperature. Therefore, the impact of tentative design changes on detailed temperature profile during synthesis process should be considered. This, in turn requires power profiles, which depend on floorplanning and power models. Even the fastest detailed thermal analysis and floorplanning algorithms cannot be included within the inner loop of synthesis without greatly reducing the solution space explored in a given amount of time. Therefore, we have proposes a two-stage synthesis process in which a potentially-slow but high-quality stochastic optimization algorithm is first used to
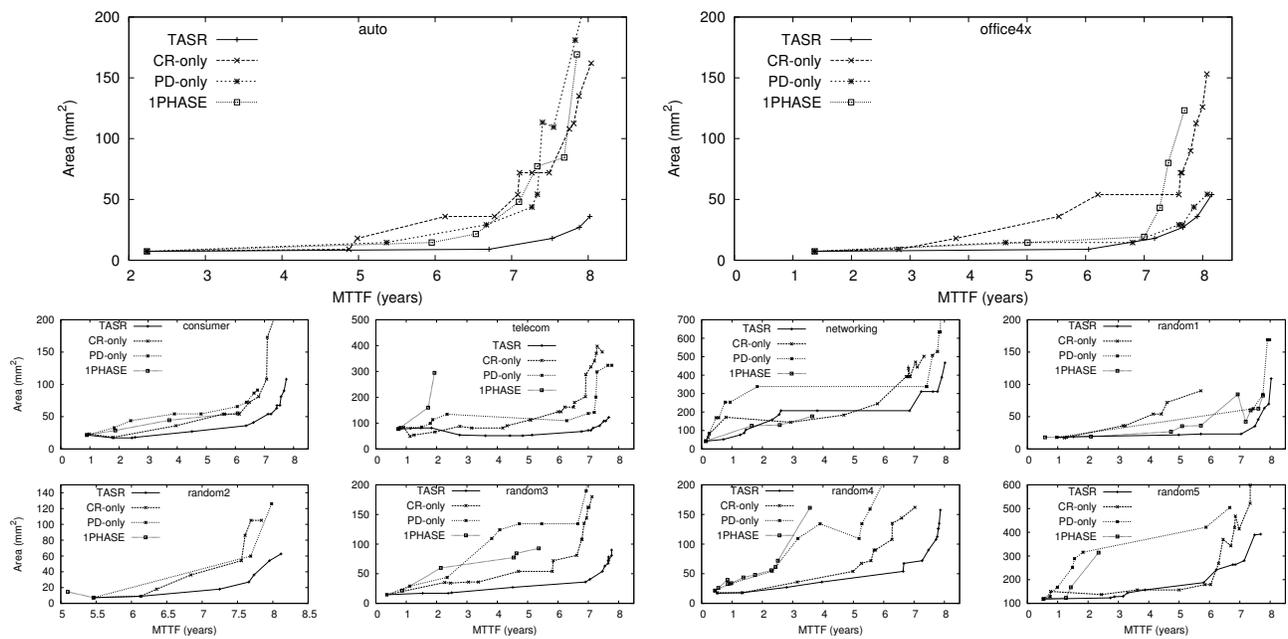
Figure 3: Comparison of different optimization heuristics.

minimize solution area. Starting from this promising location in the solution space, a reliability enhancement heuristic explores the area–MTTF tradeoff curve.

Our results indicate that this synthesis approach greatly outperforms simply adding MTTF into a stochastic optimization algorithm as another objective. The proposed synthesis flow increases MPSoC system mean time to failure by an average of 85% with less than 5% area cost and by an average of 436% with less than 25% area cost, compared to area-optimized solutions. As long as power densities remain high and the dominant lifetime failure processes remain strongly dependent on temperature, our results indicate that thermal and structural redundancy optimization during synthesis have the potential to increase MPSoC lifetime with low area cost.

5. REFERENCES

[1] "International Technology Roadmap for Semiconductors," 2006, http://public.itrs.net.

[2] J. Srinivasan, et al., "Exploiting structural duplication for lifetime reliability enhancement," in *Proc. Int. Symp. Computer Architecture*, June 2005, pp. 520–531.

[3] A. K. Coskun, et al., "Analysis and optimization of MPSoC reliability," *J. Low Power Electronics*, pp. 56–69, Apr. 2006.

[4] P. Eles, et al., "System level hardware/software partitioning based on simulated annealing and tabu search," *ACM Trans. Design Automation Electronic Systems*, vol. 2, pp. 5–32, Jan. 1997.

[5] J. Henkel and R. Ernst, "A hardware/software partitioner using a dynamically determined granularity," in *Proc. Design Automation Conf.*, June 1997, pp. 691–696.

[6] Y. Xie, et al., "Reliability-aware co-synthesis for embedded systems," in *Proc. Int. Conf. Application-Specific Systems, Architectures, and Processors*, Sept. 2004.

[7] C. Lee and S. Ha, "Hardware-software cosynthesis of multitask MPSoCs with real-time constraints," in *Proc. Int. Conf. ASIC*, Oct. 2005, pp. 919–924.

[8] K. Skadron, et al., "Temperature-aware microarchitecture," in *Proc. Int. Symp. Computer Architecture*, June 2003, pp. 2–13.

[9] Y. Yang, et al., "ISAC: Integrated Space and Time Adaptive Chip-Package Thermal Analysis," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Jan. 2007.

[10] U. Y. Ogras and R. Mărculescu, "Energy- and performance-driven NoC communication architectures synthesis using a decomposition approach," in *Proc. Design, Automation & Test in Europe Conf.*, Mar. 2005, pp. 352–357.

[11] M. Glaß, et al., "Reliability-aware system synthesis," in *Proc. Design, Automation & Test in Europe Conf.*, Apr. 2007.

[12] "Embedded microprocessor benchmark consortium," http://www.eembc.org.

[13] R. P. Dick, "Multiobjective Synthesis of Low-Power Real-Time Distributed Embedded Systems," Ph.D. dissertation, Dept. of Electrical Engineering, Princeton University, July 2002.

[14] S. Prakash and A. Parker, "SOS: Synthesis of application-specific heterogeneous multiprocessor systems," *J. Parallel & Distributed Computing*, vol. 16, pp. 338–351, Dec. 1992.

[15] J. Hou and W. Wolf, "Process partitioning for distributed embedded systems," in *Proc. Int. Wkshp. Hardware/Software Co-Design*, Mar. 1996, pp. 70–76.

[16] T.-Y. Yen, "Hardware-software co-synthesis of distributed embedded systems," Ph.D. dissertation, Dept. of Electrical Engg., Princeton University, June 1996.

[17] Z. Gu, et al., "Application-specific MPSoC reliability optimization," *IEEE Trans. VLSI Systems*, to appear.

[18] Joint Electron Device Engineering Council, "Failure mechanisms and models for semiconductor devices," in *JEDEC Publication JEP 122-B*, Aug. 2003.

[19] L. Ting, et al., "AC electromigration characterization and modeling of multilayered interconnections," in *Proc. Int. Reliability Physics Symp.*, Mar. 1993, pp. 311–316.

[20] B. Schroeder and G. A. Gibson, "A large-scale study of failures in high-performance computing systems," in *Proc. Int. Conf. Dependable Systems and Networks*, June 2006, pp. 249–258.

[21] A. Mishra and P. Banerjee, "An algorithm-based error detection scheme for the multigrid method," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 52, no. 9, pp. 1089–1099, Sept. 2003.

[22] R. P. Dick, D. L. Rhodes, and W. Wolf, "TGFF: Task Graphs for Free," in *Proc. Int. Wkshp. Hardware/Software Co-Design*, Mar. 1998, pp. 97–101.

[23] T. Sakurai, "A JSSC classic paper: The simple model of CMOS drain current," *IEEE Solid State Circuits Society Quarterly Newsletter*, pp. 4–5, Oct. 2004.