Power and Reliability Management of SoCs

Tajana Simunic Rosing, *Fellow, IEEE*, Kresimir Mihic, *Fellow, IEEE*, and Giovanni De Micheli, *Fellow, IEEE*

***Abstract*—Today's embedded systems integrate multiple IP cores for processing, communication, and sensing on a single die as systems-on-chip (SoCs). Aggressive transistor scaling, decreased voltage margins and increased processor power and temperature have made reliability assessment a much more significant issue. Although reliability of devices and interconnect has been broadly studied, in this work, we study a tradeoff between reliability and power consumption for component-based SoC designs. We specifically focus on hard error rates as they cause a device to permanently stop operating. We also present a joint reliability and power management optimization problem whose solution is an optimal management policy. When careful joint policy optimization is performed, we obtain a significant improvement in energy consumption (40%) in tandem with meeting a reliability constraint for all SoC operating temperatures.**

***Index Terms*—Optimal control, power consumption, reliability management.**

I. Introduction

TODAY's embedded systems consist of a number of heterogeneous processing, communication, and sensing components. Embedded components are increasingly being integrated into systems-on-chip (SoCs). For example, TI's OMAP2420 SoC contains a general-purpose ARM processor, a digital signal processor (DSP), graphics accelerators, video processor, four different communications processors (WLAN, WAN, WPAN, IrDA), audio and touch screen controllers, various memory interfaces, and a number of other input/output (I/O) controllers [1]. A large number of cores integrated on a single chip invariably leads to issues related to management of power consumption and temperature, both of which directly affect the SoC reliability.

Dynamic voltage scaling (DVS) reduces the power consumption by scaling down voltage and frequency of operation at run time. As power is scaled down, so is device temperature, and thus the chance of hard errors is reduced and reliability is improved. Note that overly aggressive DVS can increase the soft error rate as discussed, for example, in [34]. DVS is used during active operation of the SoC cores, and thus it does not address the issue of saving power during longer idle times.

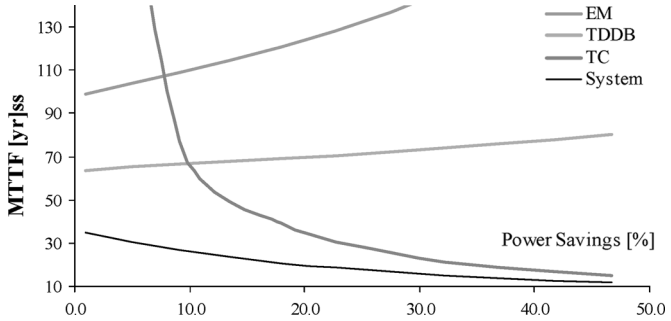

Fig. 1. Single core failures (95-nm feature size).

Dynamic power management (DPM), on the other hand, is defined as a set of policies whose aim is to save power by placing system components into low-power states when idle. Curbing power dissipation helps by lowering the device temperatures and reducing the effect of temperature-driven failure mechanisms, thus, making components more reliable. On the other hand, aggressive power management policies can decrease the overall component reliability because of the degradation effect that temperature cycles have on modern IC materials [7], [20], [23]. To illustrate this effect, in Fig. 1 we show a tradeoff between mean time to failure (MTTF) and power savings due to three most commonly modeled failure mechanisms: *electromigration* (EM), *time dependent dielectric breakdown* (TDDB), and *thermal cycling* (TC). These values are obtained from our partner silicon manufacturer for 95-nm technology as a result of their standard testing cycle. This particular test core had one sleep state, but no ability for DVS. Although more aggressive power management policies help improve MTTF due to EM and TDDB, they also have a significant cost due to thermal cycles failure mechanism, because frequent shutdowns negatively affect the TC failure rate. As a result, the overall MTTF decreases as the power savings increase. DVS can help improve reliability, but is limited to saving power and energy only when cores are active. Since a number of researchers have already studied the effect of DVS on reliability [3]–[5], in this work we focus on the tradeoff between power management (DPM) and reliability in SoCs.

There are several interesting problems that can be addressed. The first problem is to determine whether or not, for a given system topology, DPM affects reliability and to find if such an effect is beneficial or not. The second problem is to include reliability as an objective or a constraint in the policy optimization. The third problem is the combined search for system topologies and joint DPM policies to achieve reliable low-energy design. All problems involve both run-time strategies as well as design issues.

In this paper, we focus on the first two problems. The first one enables us to understand the relationship between run-time

Manuscript received June 30, 2005; revised February 22, 2006. This work was supported in part by UC Micro under Grant 06-198 and by the Swiss National Fund.

T. S. Rosing is with the Department of Computer Science Engineering, University of California at San Diego, La Jolla, CA 92093 USA (e-mail: tajana@ucsd.edu).

K. Mihic is with Stanford University, Stanford, CA 94305 USA (e-mail: kmihic@stanford.edu).

G. De Micheli is with EPFL, Lausanne 1015, Switzerland (e-mail: giovanni.demicheli@epfl.ch).

Digital Object Identifier 10.1109/TVLSI.2007.895245

power management and reliability analysis. We evaluate reliability, performance, and power consumption of computational elements (cores) in SoCs by modeling system-level reliability as a function of failure rates, system configuration and management policies. We compare and contrast savings in terms of MTTF and power due to various management policies. Our overall objective is to introduce design constraints, such as MTTF, in the design space spanned by performance and energy consumption. Another major novelty and contribution of this paper is the definition of a joint DPM and dynamic reliability management (DRM) optimization method that yields optimal system-level run-time policies. We evaluate policies on single and multicore systems. Experimental results show that with careful joint optimization we can save energy by 40% while meeting both reliability and performance constraints.

The rest of this paper begins with an overview of related work. Reliability models are introduced in Section III. The explanation of simulator functionality can be found in Section V, while the description of the optimizer is in Section IV. The results of our methodology follow in Section VI and Section VII summarizes our contributions.

II. Related Work

A number of issues related to SoC design have been discussed to date, ranging from managing power consumption, to addressing problems with interconnect design [9]–[13]. Previous work for energy management of networked SoCs mainly focused on controlling the power consumption of interconnects, while neglecting managing power of the cores. A stochastic optimization methodology for core-level dynamic voltage and power management of large SoCs with both node- and network-centric views using a closed-loop control model has been presented in [16].

Reliability of SoCs is another area of increasing concern. A good summary of research contributions that combine performance and reliability measures is given in [14]. Improving system reliability and increasing processor lifetime by implementing redundancy at the architecture level is discussed in [6] and [15]. A number of fault-tolerant microarchitectures have been proposed that can handle hard failures at performance cost [19]. The RAMP simulator models microarchitecture MTTF as a function of the failure rates of individual structures on chip due to different failure mechanisms [20]. RAMP gets activity estimates from performance simulation with an instruction level simulator and then uses them to obtain power consumption of microarchitecture components with Wattch in order to feed that information into HotSpot [3] for temperature evaluation. Once temperature is available, it can then evaluate failure rates due to each effect.

HotSpot, along with ThermalHerd [4] are used for thermal modeling and characterization. Recent approaches for dynamic thermal management (DTM) target architecture level with the goal of reducing power density in thermal hot spots by employing a number of different techniques such as global clock gating [2], migrating computation to spare units [3], traffic rerouting [4], predictive techniques for multimedia applications [5], and DVS specifically for thermal management [3]–[5]. DTM focuses on reduction of microarchitectural thermal hot spots at run time, but does not directly consider long-term reliability nor power consumption. As a result, the simulations are limited to relatively short workloads due to prohibitively long running times. In our work, we study the long term system reliability as a function of power management policies. Our optimizer gives a management policy capable of minimizing system power consumption under MTTF constraint (typically units measure in years) and performance constraint. The simulation we developed is targeted at evaluating behavior of large, multiprocessor SoCs, with workloads normally experienced during their useful life.

In our reliability analysis, we focus on hard failure mechanisms which cause irrecoverable component failures. Open interconnect line due to electromigration is an example of a hard failure. Notice that this is in contrast to soft (or transient) failure mechanisms and their effect on power consumption which have been studied by a number of researchers (e.g., [32]–[35]). An overview of most commonly observed hard failure mechanisms that affect the current semiconductor technologies is given in [22]. The effect of a temperature gradient on the electromigration failure mechanism has been investigated by many researchers, for example in [23]. Similarly, time-dependent dielectric breakdown (TDDB) has been studied extensively, an example of a model for TDDB is in [25]. Although package thermal cycles are a well-known phenomena and as such have been subject to a number of publications (see, for example, [20]), fast thermal cycles present on chips have been studied only recently. A description of the connection between fast thermal cycling and thin film cracking (interlayer dielectric, interconnections) is presented in [24] and a model is given in [32]. In contrast to previous contributions, our work presents, for the first time, a unified methodology for the joint optimization of reliability, power consumption, and performance in SoCs. In Section III, we give an overview of reliability models used in this work.

III. Reliability Modeling

Integrated systems can be abstracted by a reliability network, i.e., a connection of components labeled by their failure rates [20]. The network shows, by means of series/parallel connection of components, the conjunctive/disjunctive relations among component operating state to insure system correct operation. Failure rates, defined as the speed at which components are likely to fail, depend on the operation state of a component. Although for small designs it is possible to analytically calculate system reliability, for larger systems, such SoCs with many integrated cores, it becomes intractable and thus simulation is needed. Additionally, there is a need to develop policies for run-time power and reliability management of SoCs. An important contribution of this work is optimization of system-level power consumption under reliability and performance constraints. Second, we present a system-level simulator capable of evaluating power, reliability, and performance of large multiprocessor SoCs. To the best of our knowledge, this is the first time that reliability measures have been modeled, optimized, and simulated jointly with DPM. In this section, we present a reliability model and then outline the DPM model.

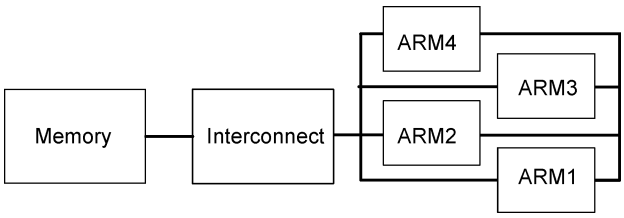

Fig. 2. Sample design.

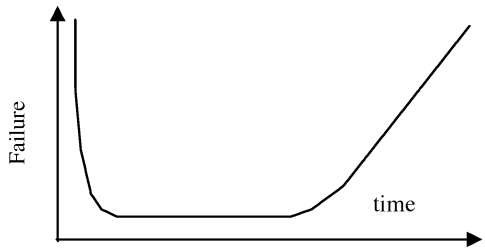

Fig. 3. Bathtub curve.

A. System Reliability

Systems are interconnections of components. We use the term *core* to refer to one of the SoC components that perform computing, storage, or communication function. From a reliability analysis standpoint, components are in *series (parallel)* if the overall correct operation hinges upon the *conjunction (disjunction)* of the correct operation of components. Fig. 2 illustrates the two configurations. The memory and interconnect form a series combination. In order to have correct system operation, both components have to be working properly. On the other hand, four ARM cores on a single die would form a parallel combination. It is enough to have one of the cores operating correctly in order for the whole system to be operational. A system is, therefore, characterized by its topology, i.e., by a *reliability graph* [26]. In this work, we use reducible graphs so the system reliability can be computed bottom-up, by considering series/parallel compositions of subsystems.

In general, failure rates are dependent on aging (time) and on temperature. Fig. 3, the bathtub curve, is a common way to show failure rates as a function of time. The curve has three distinctly different regions. Initial burn-in period and final wear-out period are typically modeled with Weibull distribution, while the failures during useful life are best described using an exponential distribution with a constant failure rate. Since we are interested in assessing the reliability over the typical operation time, we assume that the failure rates are constant in time, as shown by the middle range of the curve. Clearly, the actual value of the failure rate is a function of many parameters, some of which are temperature, power state of the component, and the frequency of switching between power states.

The reliability of a system is the probability function $R(t)$, defined on the interval $[0, \infty]$, that a system will operate correctly with no repair up to time t. The reliability is defined as a function of *failure rate*, $\lambda_f(t)$, (1). Another variable commonly used to describe the system reliability characteristic is MTTF as shown in (2). When we use a constant failure-rate model, we can represent the component reliability using exponential distribution with a failure rate, λ_f as follows: $R(t) = e^{-\lambda_f t}$, with $\mathrm{MTTF} = 1/\lambda_{\mathrm{f}}$

$$R(t) = e^{-\int_0^t \lambda_f(t')dt'} \tag{1}$$

$$\mathrm{MTTF} = \int_0^\infty R(t)dt. \tag{2}$$

Integrated systems, such as SoCs, consist of many cores connected with a complex interconnect structure. Often when a core or interconnect fails, another core or interconnect can take over its functionality. Thus, such a system has built-in redundancy. In order to model the overall system reliability, we need to define the relationship between topology, redundancy, and component power state. The system components can be organized in series and/or in parallel as shown in Fig. 2. The overall system reliability can be calculated by applying the rules for series and parallel composition, under the assumption that failure rates are statistically independent from each other. This assumption is widely used in industry [22]

$$R_{\mathrm{system}}(t) = \prod_{i-0}^{n} R_i(t) \Rightarrow R_{\mathrm{system}}(t) = e^{-\sum_{i=0}^{n} \lambda_{f_i} t}. \tag{3}$$

The system built with n series components fails if any of its components fails as shown in (3). When failure rates are constant, the failure rate of a series composition is the sum of the failure rates of each component as shown in (3). Alternatively, the parallel combination fails only if all n components that are in parallel fail

$$R_{\mathrm{system}}(t) = 1 - \prod_{i-0}^{n} \left(1 - R_i(t)\right). \tag{4}$$

Systems with parallel structures have built-in redundancy. Such systems can either have all components concurrently operating (*active parallel*) or only one component active while the rest are in low power mode (*standby parallel*). Active parallel combination has higher power consumption and lower reliability than standby parallel, but also faster response time to failure of any one component. The combined failure rate of M active components, λ_{fap}, is defined using binomial coefficient C_i^M, and active reliability rate λ_f as shown in (5). A good example of active parallel combination is when the ARM cores shown in Fig. 2 run the same safety-critical code. The fact that all four are active reduces the overall system reliability (and also significantly increases the system power consumption). Both power consumption and reliability can be improved if only one communication core is used at a time, with others saving power in standby parallel combination

$$\lambda_{\mathrm{fap}} = \sum_{i=1}^{M} (-1)^{i-1} \frac{C_i^M}{i\lambda_f}. \tag{5}$$

The failure rate of M parallel components that are all in standby is $\lambda_{\mathrm{fsp}} = \lambda_{\mathrm{fs}}/M$ [26]. To get the overall failure rate we need to combine M standby components with one active

parallel component. For example, a system that has four ARM cores, where one core is active and the others are in sleep state, the overall system failure rate would be a parallel combination of one active and three standby components. The next step is to formulate device failure rates as a function of different failure mechanisms and power states.

B. Failure Mechanisms

In this work, we focus on the reliability of components during their useful life and thus we neglect aging but we do consider temperature dependence as a function of power state and time. We assume that components can be in different operational states (e.g., *active*, *idle*, *sleep*) characterized by parameters such as voltage and frequency, which determine the component temperature. Thus, failure rates can be considered constant within any given operational state. We consider three failure mechanisms most commonly used by semiconductor industry: EM, TDDB, and TC. Although, for our work, we obtain core failure rates for each of the three failure mechanisms from our industry partner's measurements, here, we present a set of models that can be used when measurements are not available to calculate the rates.

EM is a result of momentum transfer from electrons to the ions which make interconnect lattice. It leads to opening of metal lines/contacts, shortening between adjacent metal lines, shortening between metal levels, increased resistance of metal lines/contacts, or junction shortening. The MTTF due EM process is commonly described by Black's model

$$\mathrm{MTTF}_{\mathrm{EM}} = A_o(J - J_{\mathrm{crit}})^{-n} e^{\frac{Ea}{kT}} \tag{6}$$

where A_o is an empirically determined constant, J is the current density in the interconnect, J_{crit} is the threshold current density, and k is the Boltzmann's constant, $8.62{*}10^{-5}$. For aluminum alloys E_a and n are 0.7 and 2, respectively. EM failure rate needs to be modeled for idle and active states only, because leakage current present in the sleep state is not large enough to cause the migration. In this work we formulate the EM failure rate as a product between an average measured value in a given power state s, $\lambda_{m,s}^{\mathrm{EM}}$, and a factor that is a function of temperature in that state T_s. Temperature is calculated during simulation according to equations shown in Section V

$$\lambda_{\mathrm{core},s}^{\mathrm{EM}} = A_o'(J_s - J_{\mathrm{crit}})^n e^{\frac{-Ea}{kT_s}} = \lambda_{m,s}^{\mathrm{EM}} e^{\frac{-Ea}{kT_s}} \ \forall s = \text{active,idle.} \tag{7}$$

TDDB is a wear out mechanism of dielectric due electric field and temperature. The mechanism causes the formation of conductive paths through dielectrics shortening the anode and cathode. MTTF due to TDDB can be defined with the field-driven model

$$\mathrm{MTTF}_{\mathrm{TDDB}} = A_o e^{-\gamma E_{\mathrm{ox}}} e^{\frac{Ea}{kT}} \tag{8}$$

where A_o is an empirically determined constant, γ is the field acceleration parameter, and $\mathrm{E_{ox}}$ is the electric field across the dielectric. The activation energy E_a for intrinsic failures in $\mathrm{SiO_2}$ is found to be 0.6–0.9 and for extrinsic failures about 0.3 [22]. The failure rate due to TDDM mechanism for active, idle, and sleep state can be defined much in the same way as for the EM mechanism as a product between an average measured value in a given power state s, $\lambda_{m,s}^{\mathrm{TDDB}}$, and a factor that is a function of temperature in that state T_s. Again, the temperature is calculated dynamically during simulation as described in Section V

$$\begin{aligned}\lambda_{\mathrm{core},s}^{\mathrm{TDDB}} &= A_o' e^{\gamma E_{\mathrm{ox},s}} e^{\frac{-Ea}{kT_s}} \\ &= \lambda_{m,s}^{\mathrm{TDDB}} e^{\frac{-Ea}{kT_s}} \ \forall s = \text{active,idle,sleep.}\end{aligned} \tag{9}$$

TC can induce plastic deformations of materials that accumulate every time the cycle is experienced. This eventually leads to creation of cracks, fractures, short circuits, and other failures of metal films and interlayer dielectrics as well as fatigue at the package and die interface. The effect is caused by the large difference in thermal expansion coefficients between metallic and dielectric materials, the silicon substrate, and the package.

Thermal cycles are the part of normal operating conditions such as power up and down, or going into low power or standby mode. The effect of low frequency thermal cycles (like on/off) has been well studied by packaging community and used extensively in the qualification process. Such cycles affect the package and die interface mostly and are well modeled by the Coffin–Manson model

$$N_f = C_o|\Delta T - \Delta T_o|^{-q} \tag{10}$$

where N_f is the number of thermal cycles to failure, C_o is a material dependent constant, ΔT is the temperature cycle range, ΔT_o is the portion of the temperature cycle range in the elastic region, and q is an empirically determined Coffin–Manson exponent. Commonly used values for q are 1–3 for ductile metal, 3–5 for hard metal alloys and intermetallics, and 6–9 for brittle fracture [31].

Thermal cycles that occur with higher frequencies, for example due to power management, are gaining in importance as features sizes get smaller and low-k dielectric is introduced to the fabrication process [29]. Recent work [32] shows that such cycles play a major role in cracking of thin film metallic interconnects and dielectrics (brittle materials). Expected number of thermal cycles before core failure is given in (11). It does not only depend on the temperature range between power states $(T_{\max} - T_{\min})$ but is also strongly influenced by the average temperature in the sleep state, $T_{\mathrm{avg},s}$ and the molding temperature of the package process, T_{mold}. The exponent q ranges from 6–9, and $C_{1,2}$ are fitting constants defined in [32] for on-chip structures. Mechanical properties of the interlayer dielectric layers are very dependent on the nature of the processing steps. As a result, when $T_{\mathrm{avg},s}$ increases, the stress buildup on the silicon due the package decreases resulting in a longer lifetime

$$N_f = C_o\left[C_1(T_{\max} - T_{\min}) - C_2(T_{\mathrm{avg},s} - T_{\mathrm{mold}})\right]^{-q} \tag{11}$$

$$\lambda_{\mathrm{core},s}^{\mathrm{TC}} = C_o'\left[C_1(T_{\mathrm{active}} - T_{\mathrm{sleep}}) - C_2(T_{\mathrm{avg},s} - T_{\mathrm{mold}})\right]^q f_s. \tag{12}$$

For a power managed core, two distinct thermal cycle loops exist. The first one is between active and idle states during normal operation. As it occurs very between states which only have a small difference in power consumption, the temperature

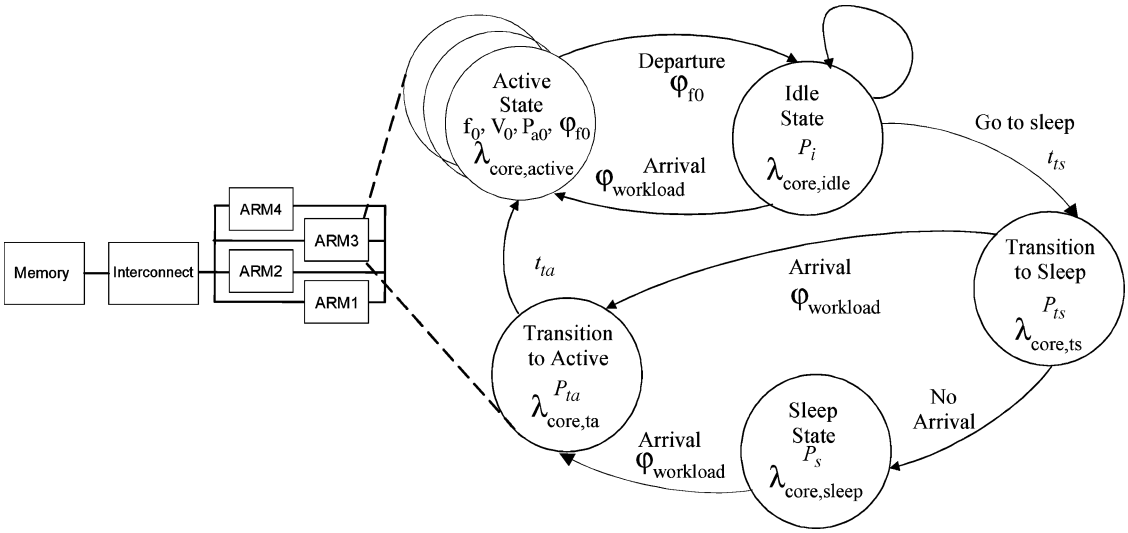

Fig. 4. System model.

differences are relatively small and thus unlikely to cause long-term reliability problems. As a result, we do not model these cycles. On the other hand, we do include cycles which happen when a core transitions between active and sleep states. These cycles occur between states that have a large difference in power consumption, and thus are likely to have a significant temperature difference that in turn causes reliability problems. Therefore, we can calculate the total failure rate due to the thermal cycling effect as shown in (12). $T_{\text{ave},s}$ and T_{mold} are the average temperature in sleep state and the temperature of package molding process, while Cs are fitting constants. T_{active} and T_{sleep} are current temperatures in the active and sleep states and f_s is the frequency of transitioning into the sleep state. Current temperature values are obtained during simulation as described in Section V.

C. Core Failure Rate

The standard model used by the industry for system level reliability is the *sum-of-failure-rates* (SOFR) model. The model implies that the core is a series failure system, and as such can be represented as the sum of the failure rates of individual failure mechanisms, which are commonly assumed to be statistically independent from each other [22]. Thus, failure rate of a power managed core $\lambda_{\text{core\_}pm}$ can be found by summing up failure rates of individual failure mechanisms $\lambda_{\text{core}} = \lambda_{\text{core}}^{\text{EM}} + \lambda_{\text{core}}^{\text{TDDB}} + \lambda_{\text{core}}^{\text{TC}}$. While EM and TDDB failure mechanisms improve when power management is used since the overall system temperature is lower, fast thermal cycles can cause the overall failure rate to grow due to too frequent switching between the power states. In fact, as each core's temperature changes due to changes in the workload or during power state transitions, the failure rates due to EM, TDDB, and TC also change. In general, analytical formulae can be used to calculate the overall system MTTF only for relatively simple configurations. Simulation is needed in order to accurately model reliability changes of more complex power managed SoC configurations. Furthermore, neither analytical formulae nor simulation can answer how to design a management policy that minimizes power consumption while meeting MTTF constraints. Thus, in the following sections, we first describe our optimization methodology, followed by the simulation platform we developed to evaluate the tradeoffs between power management and reliability.

IV. Power and Reliability Optimization

In this section, we describe a method for finding the optimum power management policy with reliability and performance constraints. We define the optimization problem given a system topology (e.g., cores shown in Fig. 4) and a set of component operational states characterized by failure rate, power consumption, and performance. The result of optimization is a power and reliability management policy.

A policy manager makes decisions at each event occurrence (e.g., arrival or departure of a request). The *interevent time set* is defined as $T = \{t_i; \text{s.t. } i \in 0, 1, 2, \ldots, i^{\max}\}$ where each t_i is the time between two successive event arrivals and $i_{\max}$ is the index of the maximum time horizon. We denote by $s_i \in S_i$ the system state at decision epoch i. Commands are issued whenever the system state changes. We denote by $a_i \in A_i$ an action (or command) that is issued at decision epoch i. An example of a command would be to transition to sleep state from idle state. In general, commands given are functions of the state history and the policy. In the specific case where all transitions between states can be modeled using exponential distribution, the policy optimization problem can be formulated using simple Markov decision processes (e.g., see [17]) and the decisions, and states associated with them are not a function of state history (and time). Examples of such states are active and sleep state with pending requests. On the other hand, when more than one state transition is not exponentially distributed, then time indexing is needed, as shown for idle and sleep with empty queue states. In that case, the *time-indexed semi-Markov decision processes* (TISMDP) model is used, as described in more detail in [18].

Each core in the reliability network is modeled with a *power and reliability state machine* (PRSM) as shown for the ARM core in Fig. 4. PSRM is a state diagram relating service levels to the allowable transitions among them. Multiple cores form

TABLE I
SYSTEM MODEL CHARACTERISTICS

State	Distribution	Parameters	Failure Rate
Active	Exponential	$\varphi_{\text{core}, fV}$,	$\lambda_a = \lambda_a^{EM} + \lambda_a^{TDDB}$
		$\varphi_{workload}$	
Idle	Any	*Collected Data*	$\lambda_i = \lambda_i^{EM} + \lambda_i^{TDDB}$
Transition	Uniform	t_{min}, t_{max}	
Sleep	Any	*Collected*	$\lambda_s = \lambda_s^{TDDB} + \lambda_s^{TC}$
Queue = 0		*Data*	
Queue > 0	Exponential	$\varphi_{workload}$	

a reliability network of series and parallel combinations of single core PRSM models. Single core PRSM characterizes each state by its failure rate $\lambda_{\text{core,state}}$ and power consumption P_{state}. Thus, the active state i is characterized by the failure rate $\lambda_{\text{core,activei}}$, frequency, and voltage of operation f_i, V_i, which is equivalent to the core processing rate φ_{fi}, and power consumption P_{ai}. In the active state, the workload and core's data processing times follow the exponential distribution with rates $\varphi_{\text{workload}}$ and $\varphi_{\text{core\_}fi}$. In the idle state, a core is active but not currently processing data. The sleep state represents one or more low power states a core can enter. *TransitionToSleep* and *TransitionToActive* states model the time and power consumption required to enter and exit each sleep state. Transition times to/from low-power states follow uniform distribution with average transition times t_{ts}, t_{ta} [12]. In the active state, the power manager decides the appropriate frequency and voltage setting, where as in the idle state the primary decision is which low-power state core should transition to and when the transition should occur. The arcs represent transitions between the states with the associated transition times and rates. Table I summarizes all distributions used in modeling performance, power consumption, and failure rates. Please note that these choices of distributions have been validated by experimental measurements [18]

$$E[T_{\text{state}}] = (T_{\text{active}} - T_{\text{state},ss})e^{-\frac{y(s,a)}{\tau}} + T_{\text{state},ss}. \quad (13)$$

Failure rates change with each power state since different level of power consumption causes a different temperature. We calculate the expected temperature for each state ($E[T_{\text{state}}]$) in (13) as a function of the expected time spent in that state $y(s,a)$ the steady-state temperature for the state (which is directly determined by state's power consumption) and the technology parameter ($\tau \approx c\rho a^2$, defined in more detail in Section V). Using the expected temperature, we can calculate a stationary value for failure rates due to each mechanism and per each power state. Thus, the assumption that all processes used to formulate the optimization problem are stationary continues to hold. Note that as any of the parameters change, for example, the expected workload arrival rate, the optimal policy just needs to be recalculated using the new values.

With the power management optimization model from Fig. 4, the state transition distributions and reliability rates outlined in Table I, we can formulate a linear program (LP) for the minimization of energy consumption under reliability constraint as shown in (14). The first constraint is known as a balance equation since it requires that a number of entries into a given state to equal the number of exists out of that state. The second constraint limits the sum of all probabilities to equal one. Each probability is represented by a product between the expected time spent in a state s under a command a, $y(s,a)$, the unknowns of the LP $f(s,a)$. The $A \times S$ unknowns in the LP $f(s,a)$, called *state-action frequencies*, are the expected number of times that the system is in state s and command a is issued. The last three equations specify constraints on performance and reliability. The result of optimization is a globally optimal power management policy that is both stationary and randomized. The policy can be compactly represented by associating a probability of issuing command a when the system is in state s, $x(s,a) = f(s,a)/\Sigma f(s,a')$ with each state and action pair. We explain now in more detail all variables used in LP formulation starting with the reliability constraint. Both continuous and time indexed versions are presented, labeled with dt and Δt, respectively.

The reliability constraint, Tpl is a function of the system topology, i.e., $Tpl = f(\text{series, parallel combinations})$. For example, with series combinations $Tpl = \sum \lambda_{\text{core},s}$, and with parallel standby $Tpl = \lambda_{\text{core,standby}}/N_{\text{standby}}$

$$\begin{aligned}
\min \quad & \sum_{c=1}^{N} \cos t_{\text{energy},c} \\
\text{s.t.} \quad & \sum_{a\in A} f(s,a) - \sum_{a\in A}\sum_{s'\in S} m(s'|s,a) f(s',a) = 0; \ \forall s, \forall c_s \\
& \sum_{a\in A}\sum_{s\in S} y(s,a) f(s,a) = 1; \quad \forall c_s \\
& \sum_{c=1}^{N} \cos t_{\text{perf},c} < \text{Perf}_{\text{const}}; \quad \forall c \\
& Tpl(\lambda_c) \le \text{Rel}_{\text{const}}; \quad \forall c_s \\
& \lambda_c = \sum_{i\in F}\sum_{a\in A}\sum_{s\in S} \lambda_{\text{core}}^{i}(s,a) y(s,a) f(s,a). \qquad (14)
\end{aligned}$$

A reliability network may have a number of series and parallel combinations of cores. Each core's failure rate, λ_c, is a sum of failure mechanisms, $i \in \{\text{EM, TDDB, TC}\}$, when the core is in the state s and the action a is given. For example, the reliability constraint is given in (15) for a core that has one *active* (A), *idle* (I), and *sleep* (S) state and two actions: *go to sleep* (S) and *continue* (C). Failure rate in each state, $\lambda_{core,state}$, is a sum of failure rates due to failure mechanisms active for that state as described in Section III

$$\begin{aligned}
\lambda_A y(A,C) f(A,C) + \lambda_I y(I,C) f(I,C) + \lambda_I y(I,S) f(I,S) \\
+ \lambda_S y(S,C) f(S,C) \le \text{Rel}_{\text{const}}. \quad (15)
\end{aligned}$$

Our model also defines two cost metrics, energy and performance. The average cost incurred between two successive events as defined in (16) is a sum of the lump sum cost $k(s_i, a_i)$ incurred when action a_i is chosen in state s_i, in addition to the

cost in state s_{i+1} incurred at rate $c(s_{i+1}, s_i, a_i)$ after choosing action a_i in state s_i. The value of cost rate, $c(s_{i+1}, s_i, a_i)$ is power consumption in a state s_i for the calculation of energy cost, and performance penalty for the performance constraint [see (16) at the bottom of the page].

When action a_i is chosen in system state s_i, the probability that the next event will occur by time t_i is defined by the cumulative probability distribution $F(t_i|s_i, a_i)$. For example, in the active state the cumulative distribution of request departures is given by $F(\text{active}, \text{departure}) = 1 - e^{-t\lambda_{\text{core}}}$. The probability that the system transitions to state s_{i+1} at or before the next event t_i is given by $p(s_{i+1}|t_i, s_i, a_i)$. For example, in active state the probability of departure is given by $p(\text{idle}|t, \text{ active, departure}) = \lambda_{\text{core}}/(\lambda_{\text{core}} + \lambda_{\text{workload}})$. In time-indexed idle and sleep states the probability of getting an arrival in time increment Δt can be calculated as follows:

$$p(s_{i+1}|t_i, s_i, a_i) = \frac{F(t_i + \Delta t) - F(t_i)}{1 - F(t_i)}. \tag{17}$$

The expected time spent in each state s when a command a is given, $y(s, a)$, is defined in (18). For example, the expected time spent in the active state is equal $1/(\lambda_{\text{core}} + \lambda_{\text{workload}})$

$$y(s_i, a_i) = \begin{cases} \int_0^\infty t \sum_{s_{i+1} \in S_{i+1}} p(s_{i+1}|t_i, s_i, a_i) F(dt|s_i, a_i) & \forall dt \\ \int_{t_i}^{t_i + \Delta t} \frac{(1 - F(t))dt}{1 - F(t_i)} & \forall \Delta t. \end{cases} \tag{18}$$

Finally, the probability of arriving into state s_{i+1} given that the action a_i was taken in state s_i is defined by

$$m(s_{i+1}|s_i, a_i) = \int_0^\infty p(s_{i+1}|t_i, s_i, a_i) F(dt|s_i, a_i). \tag{19}$$

For the time indexed states the probability of arriving to the next idle state is defined to be $m(s_{i+1}|s_i, a_i) = 1 - p(s_{i+1}|t_i, s_i, a_i)$ and of transition back into the active state is $m(s_{i+1}|s_i, a_i) = p(s_{i+1}|t_i, s_i, a_i)$.

Equations (16)–(19) are sufficient to calculate all variables needed for the LP shown in (14). The LP can then be solved using any linear program solver in a matter of less than a second. The final output of optimization is a table that specifies probabilities of transitioning cores into each of their low-power states. For example, for a StrongARM core the table might specify that the probability of going to sleep when idle for 50 ms is 30%, at 100 ms is 65%, while at 300 ms it is 100%. Such policy is implemented as follows. Upon entry to each decision state, a pseudo-random number RND is generated. The core transitions into low-power state at the time interval for which the probability of going to that state as given by the policy is greater than RND. Thus, the policy can be viewed as a randomized timeout. The core transitions into the active state if the request arrives before entry into low-power state. Once the core is in the sleep state, it stays asleep until the first request arrives, at which point it transitions back into the active state.

V. Simulation Platform

The simulator we built, as far as we know, is the first one to unify voltage scaling, power management, and reliability at the system level instead of at microarchitecture [4], [20] or lower levels. As a result, it does not require detailed models of core microarchitecture, which often is not available due to IP protection, nor does it need instruction-level simulation information in order to estimate switching activity and power consumption, and thus is significantly faster. As such it is ideal for simulating power-performance-reliability tradeoffs for large multicore systems with dynamically changing workloads that describe well SoC functionality over system's useful lifetime (typically in terms of years). In contrast to both optimization and the analytical models, the simulator also supports nonstationary workloads, thus enabling a more accurate study of DVS/DPM/DRM tradeoffs. Statistical simulation has been used extensively in the past for evaluating reliability of larger systems [36]. This is the first time that statistical simulation techniques are used for evaluation of power, performance, and reliability of SoCs.

The simulator consists of two tightly integrated components: a power management part that estimates and implements DVS and DPM policies, and a reliability part that monitors and updates the reliability network and returns the current reliability of the simulated system. Each core is modeled as a *power and reliability state machine* (PRSM) as shown in Fig. 4. Transitions between states occur due to a DVS policy (between different active states), or due to DPM policy (between idle and sleep states) or because of natural core operation, such as arrival of a workload request (arrival arcs) finishing processing on data (departure arcs) or a failure of the core.

The simulator pseudocode is given in Fig. 5. First, the simulator checks for any core failures due to various failure mechanisms modeled (e.g., EM, TDDB, TC). Each core's failure rates are calculated from the data collected at runtime (e.g., amount of time spent in a state, frequency of state changes) and core's specifications (e.g., power, voltage, transition times) using equations given in Section III. The temperature in a state is estimated using the base active state temperature T_{active}, the time spent in a state s due to an action (i.e., an instruction to perform a transition) a, $t(s, a)$, and state's steady-state temperature $T_{\text{state},ss} = T_{\text{active}} P_{\text{state}}/P_{\text{active}}$

$$T_{\text{state}} = (T_{\text{active}} - T_{\text{state},ss}) e^{-\frac{t(s,a)}{\tau}} + T_{\text{state},ss}. \tag{20}$$

$$\text{cost}(s_i, a_i) = \begin{cases} k(s_i, a_i) + \int_0^\infty \left[F(du|s_i, a_i) \sum_{s_{i+1} \in S_{i+1}} \int_0^u c(s_{i+1}, s_i, a_i) p(s_{i+1}|t_i, s_i, a_i) dt \right] & \forall dt \\ k(s_i, a_i) + \sum_{s_{i+1} \in S_{i+1}} c(s_{i+1}, s_i, a_i) y(s_i, a_i) & \forall \Delta t \end{cases} \tag{16}$$

```
loop
 select the event with the earliest time stamp
 if current time stamp < simulation length
    find the core assigned to the event
    check if core failed:
         calculate the current failure rates
         determine for each core
              if the core failed
                 search the redundancy network
                  if path found - activate it.
                       Remove failed path
                       Update redundancy net.
    end check core
    calculate system power, perf., reliability
    apply power management policy
      set all core's power/performance states
      find the next event type and duration
 else
    finish the simulation and print out data
end loop
```

Fig. 5. Simulator pseudocode.

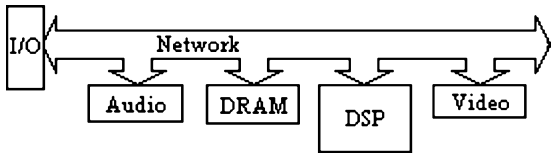

Fig. 6. SoC.

TABLE II
SoC PARAMETERS

IP block	P_{active} [W]	P_{idle} [W]	P_{sleep} [W]	t_{ts} [s]	t_{ta} [s]
DSP (TMS6211) [27]	1.1	0.5	0.01	250u	100n
Video (SAF7113H) [28]	0.44	0.44	0.07	110m	0.9
Audio (SST-Melody-DAA) [29]	0.11	0.003	3e-4	6u	0.13
I/O (MSP43011x2) [30]	1e-3	N/A	6e-6	100n	6u
DRAM (Rambus 512M) [31]	1.58	0.37	1e-2	16n	16n

The base active state temperature, shown in (21), is defined using the thermal resistances of die and package, $\mathrm{R_{thdie}}$ and $\mathrm{R_{thpackage}}$, for a reference frequency and voltage of operation in the active state f_0V_0. Thermal *RC* constant, $\tau \approx c\rho a^2$ has $c = 10^6$ J/m^{3}K for silicon thermal capacitance, $\rho = 10^{-2}$ mK/W thermal resistivity [21] and the wafer thickness a of 0.1–0.6 mm. In the future version of our system simulator, we plan to extend it with a more sophisticated spatial model of thermal effects, likely by integrating with a simulator such as HotSpot [3]

$$T_{\mathrm{active}} = P_{\mathrm{active}}(R_{th\ \mathrm{die}} + R_{th\ \mathrm{package}}). \tag{21}$$

If one or more cores fail at a particular event occurrence, then spares are identified and activated. At this time, the simulator handles both active and standby redundancy models and alters the reliability network to accommodate for failed component(s). Once the reliability network is updated, the values of the overall system power consumption, performance, and reliability are calculated for the current time period. The next step is to evaluate the management policy and schedule next sets of events, including events that result from power management commands (e.g., transitioning a core to sleep). Of all the possible next events, the simulator selects the event that happens first and continues the simulation loop. Simulating dynamic power management and reliability with one tool enables us to observe correlation and dependency between power management of multicore systems and the overall system reliability.

VI. RESULTS

In this section, we discuss the advantages and disadvantages of simulation versus optimization, in addition to studying in great detail the tradeoffs between DVS, DPM, and DRM. We use a multiprocessor SoC shown in Fig. 6 for most of our experiments. Each core's power and performance characteristics come from the datasheets [27]–[31] and are summarized in Table II. The cores support multiple power modes (*active*, *idle*, *sleep*, and *off*). Transition times between active and sleep state are defined by t_{ts} and t_{ta}. Reliability rates for each failure mechanism (EM, TDDB, TC) are based on measurements obtained for 95-nm technology. The failure rates are scaled depending on the temperature of the core, which is directly affected by the core's power state and the workload. Details of failure rate calculations have been discussed in Section III-B. Each of the cores in the system is designed to meet MTTF of ten years. Core's workload and data consumption rates ($\varphi_{\mathrm{workload}}$ and $\varphi_{\mathrm{core\_}fi}$) are extracted from a data trace collected during one day (12 hr) of typical usage. The trace consists of standard applications—MPEG4 video, MP3 audio, WWW, e-mail, telnet.

A. Optimization Results

The optimizer's objective is to determine a power and reliability management policy that minimizes power consumption under MTTF and performance constraints. Inputs to the optimizer are power, reliability, and performance characteristics of each core, along with a reliability network topology. The output is a management policy obtained from state-action frequencies $f(s, a)$ which are the unknowns of (14). The optimization results have been successfully validated against simulation—the difference was less than 1% for each design considered in this paper.

We optimize the power consumption of each core presented in Table I while keeping the minimum lifetime requirement at ten years. The objective is to observe how cores built using the same 95-nm technology and with comparable area but different power consumption respond to DPM. Optimization is performed at two internal chip temperatures (50 and 90 °C) in order to set the die operating points close to those defined in datasheets [27]–[31]. The optimization results for maximum power savings achievable at a specified temperature given MTTF constraint of ten years are shown in Fig. 7. At 50 °C most of the cores react positively to DPM and allow the maximum power savings to be achieved. When active core temperature increases to 90 °C, Fig. 7 shows that maximum power savings achievable under MTTF constraint decrease. due to thermal cycles for DSP,

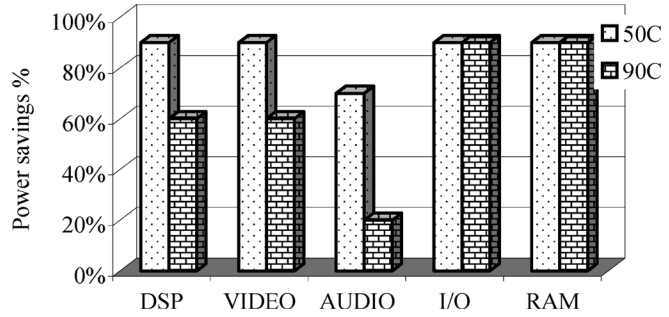

Fig. 7. Optimization of individual cores.

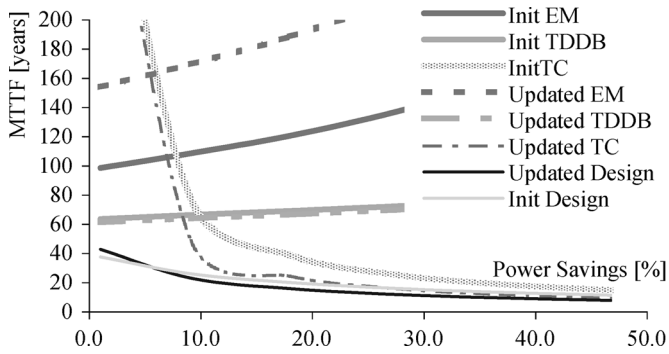

Fig. 8. Initial versus updated width of metal lines in a core.

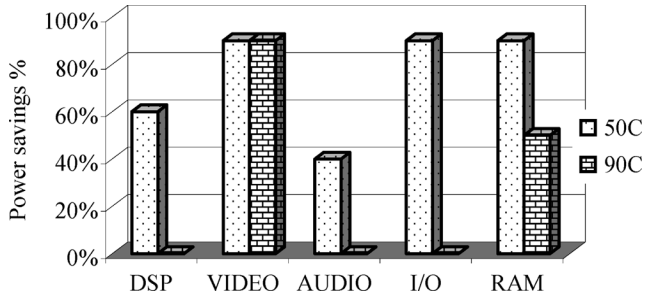

Fig. 9. Standby sleep redundancy model.

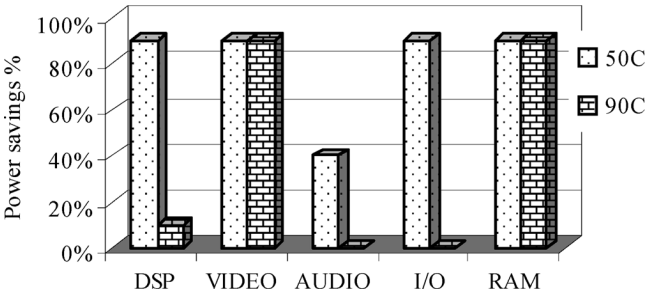

Fig. 10. Standby off redundancy model.

Video, and Audio cores. One way to try to address this problem is by redesigning cores.

Influencing the lifetime of power managed core by means of changing the design is a matter of finding the equilibrium between related physical parameters. Fig. 8 shows results of design updates done to the Video core. EM failure rate is lowered (thus, MTTF due to EM grows) by widening critical metal lines. Core area expanded by 5%, current density dropped by 20%, and the core temperature dropped by 2%. Although EM failure rate decreases significantly from the design change, the TC failure rate increases sufficiently to actually worsen the net reliability by 10%. Thus, changing a design parameters without carefully studying via simulation and optimization the overall effect the change has on all failure mechanisms might actually result in the overall degradation instead of improvement to the lifetime requirement.

Now we examine the influence of redundant components to the overall system reliability. We use the SoC shown in Fig. 6 with the core parameters given in Table II. Since all cores are essential to the correct SoC operation, the initial reliability network is their series combination. Unfortunately, although each core meets MTTF requirement of ten years, the overall system does not. To mitigate this problem, we use two types of redundancy: standby sleep configuration, where currently unused cores are in a sleep state until needed, and standby off, with unused cores turned off. Since typical embedded systems do not use all of the computational resources available in SoC at all times, it is likely that some resources are at least part of the time in a low power state, and thus might be available when a failure occurs. Figs. 9 and 10 show the maximum power savings achievable per each core assuming system MTTF of ten years. Clearly, the best power savings are with the standby off model. However, this model also has the largest wakeup delay for the unused components. The standby sleep model, shown in Fig. 9, gives more moderate power savings but has faster activation time. Results for both models show that not all cores can operate reliably at the highest temperature (e.g., no power savings for AUDIO core at 90 °C show that the system reliability constraint of ten years is not met). When we allow additional spares for DSP, AUDIO, and I/O in standby off mode, then the overall system meets MTTF of ten years while getting power savings of 40%.

B. Simulation Results

The simulator enables us to observe the system reliability complex reliability networks, under variable workloads and power management policies. For example, Fig. 11 shows that indeed, the value of system reliability is affected by the variability in the workload as the core's video stream changes from 15 frames/s for the first 300 h, and to a high definition video signal running at 60 frames/s. Optimization assumes that all distributions are stationary, so for a significant workload rate change such as the one shown in Fig. 11, the policy would have to be recalculated as soon as a change in rate is detected.

Although for very simple problems we may be able to evaluate reliability analytically using equations outlined in Section III, for more realistic cases, simulation (and optimization) is a must. The simulation results have been checked against the analytical reliability models for a simple single core design. We have also compared the MTTF values we obtained from simulations for a single core over a variety of DPM policies and system conditions with the results of testing done on the test chips designed in 95-nm technology and produced by our industry collaborators. The comparison is done by entering the test core characteristics (power states, failure mechanism constants, etc.) into the simulator, applying

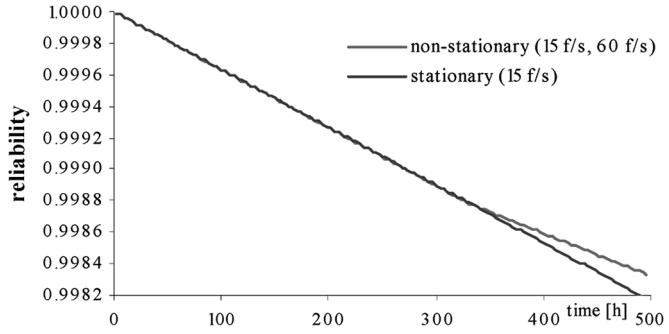

Fig. 11. Variable workload.

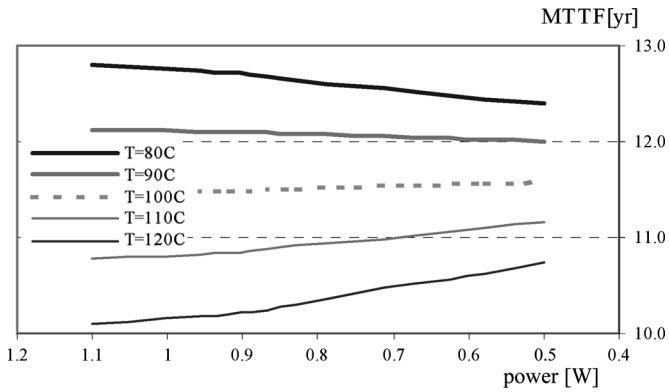

Fig. 12. MTTF of DSP core.

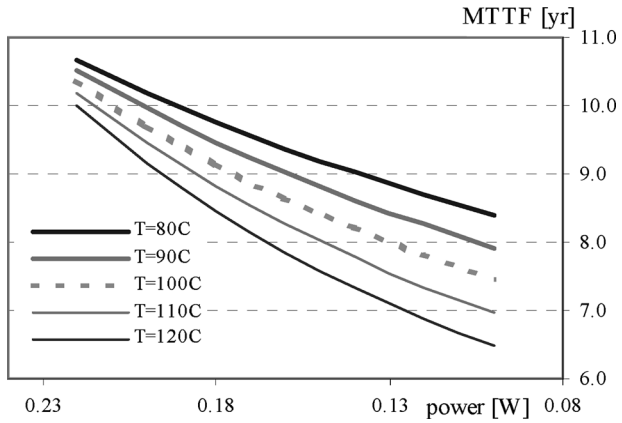

Fig. 13. MTTF of video core

a DPM policy and the workload to the core and then obtaining the MTTF over its useful lifetime. Similarly, we implement the same DPM policy and workload to the actual test core and run it through the standardized tests used to obtain the values of failure rates to report to the customers. We found that the difference between simulation and measurement is less then 15%. We next show simulation results for both single and multicore systems. Power and performance characteristics of cores used in our simulations are shown in Table II.

Single Core System: We simulated the effect of power management on two single core systems. The objective is to observe how two cores based on the same technology and of the same area, but with different power/performance characteristics, respond to DPM policies designed to minimize power consumption without considering reliability. Fig. 12 shows the simulation results of the DSP core in terms of MTTF and power consumption. On the lower range of active state temperatures (80–90 °C) the core becomes less reliable as the DPM policy changes from no power management ($P = 1.1$ W) to aggressive DPM ($P = 0.5$ W) due to a large increase in TC failure rate. At higher operating temperatures, the core becomes more reliable as gains in reliability due to EM and TDDB are more significant than losses due to TC. The results in Fig. 13 show the overwhelming influence of thermal cycles for every combination of active state temperatures and PM policies applied to the video core. The large temperature difference between sleep and active state strongly limit core reliability. These results show that it is critical to consider reliability and power management jointly: the system designed to operate for ∼10 years under constant stress fails to meet the requirement once power management is introduced (see Fig. 13).

TABLE III
XSCALE POWER STATE CHARACTERISTICS

State	Active (mW)	Idle (mW)	Freq (MHz)
P1	925	260	624
P2	747	222	520
P3	279	129	208
P4	116	64	104
Psleep	0.163	0.163	0

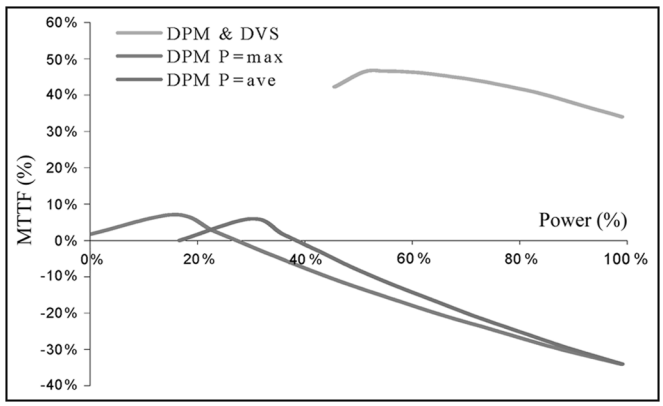

Fig. 14. Power and MTTF with DVS and DPM on XScale.

DVS is frequently suggested as a good way to manage both power and reliability since it reduces thermal hot spots at run time. Although we found that core reliability indeed does improve when DVS is used (note , we use only rated frequencies of operation; soft errors due to very aggressive DVS are not considered here), the possible power savings are less than could be obtained when DVS and DPM are used jointly. A system that implements only DVS, stays in the idle state regardless of the length of the idle period, instead of going to sleep. We illustrate the tradeoff between DVS and DPM on an example of the Intel's XScale processor PXA270 [8]. We use XScale power state characteristics given in Table III with the failure rate parameters obtained from the tests done on a set of test cores designed in 95-nm technology by our industry partner. The actual failure rates are calculated using equations given in Section III by combining parameters from measurements and temperature values given at each step in the simulation. In our simulations,

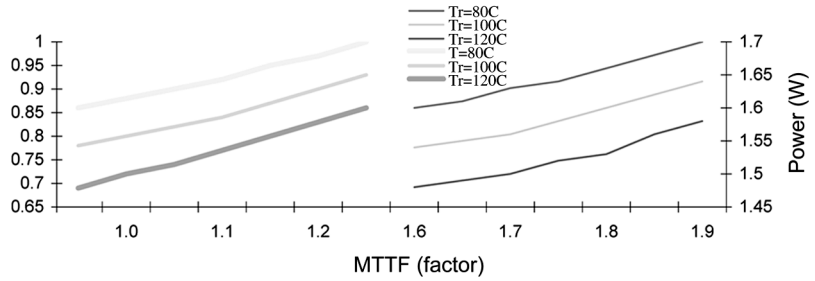

Fig. 15. System with no redundancy versus with redundancy.

TABLE IV
POWER SAVINGS AND MTTF INCREASE FOR XSCALE

Policy	Power	MTTF
None	0%	0%
DVS	35%	42%
DPM (Rmax)	16%	6%
DPM (ave)	47%	-12%
DPM (Pmax)	99%	-34%
both (Rmax)	46%	47%
both (ave)	61%	45%
both (Pmax)	99%	34%

we use one sleep state and four frequency settings for active and idle states, for a total of eight additional states.

The workload is obtained by collecting a data trace during one day (12 h) of typical usage. The trace consists of standard applications—MPEG4 video, MP3 audio, WWW, e-mail, telnet. In the simulator, we implemented the "ideal" DVS policy—the policy sets the best voltage/frequency setting for each application as determined by prior analysis. In this way, we compare as the best possible scenario with DVS relative to a more realistic situation with DPM policies. DPM policy is obtained by minimizing system energy consumption under the performance constraint [we solve the LP shown in (14) with no reliability constraint]. Table IV shows the percent of power savings and the percent increase in mean time to failure (MTTF) for four categories cases: no power management, only DVS, only DPM, and both. We also show results for DPM when reliability improvement is maximized (Rmax), power savings are maximized (Pmax), and the average case (ave). DVS gives reasonable power savings 35%, with 42% improvement in MTTF. DPM has much larger power savings, but also causes an average 12% decline in MTTF due to on chip thermal cycles resulting from DPM.

Fig. 14 shows a tradeoff of DVS and DPM for various power management policies, ranging from no DPM to a very aggressive policy (largest power savings). The best power savings and improvement in reliability are when DVS and DPM are combined $(\mathrm{DVS}, \mathrm{PM})$. Interestingly, when no DVS is used the difference in improvement in terms of MTTF is almost negligible between the two cases of DPM—the one where active state frequency of operation is set to maximum $(P = \max)$ and the one where it is set to average $(P = \mathrm{ave})$. In addition, Fig. 14 shows that there is a clear optimal point in terms of MTTF as a function of DPM. Thus, there is a need to optimize SoC reliability along with power consumption and performance.

Multicore System: We next examine the influence of redundant components to the overall system reliability and power consumption by focusing on audio and video cores (see Fig. 6 and Table II for specs). If the network has no spare components, then the entire system fails when one of the cores fails. Alternatively, either hardware redundancy can be integrated by including redundant cores (or memory banks in the case of memory), or the computation of a failed core can be mapped onto one of the other available cores but with a significant performance degradation. In our case, the video core is processing an NTSC signal while the audio core processes a 44-k, 16-bit signal. The left three curves in Fig. 15 represent the original system response consisting of one audio and one video core, while the ones on the right are for the system with redundancy. Both systems are plotted as a function of three different base active state temperatures represented with T for nonredundant system, and T_r for redundant system. We use $T_{\mathrm{active}} = 80\,^{\circ}\mathrm{C}$, no redundancy and no power management as the reference point in the figure and express all other MTTF results factor relative to the reference point (e.g., 0.5 means that the MTTF has been halved as compared to the reference point). Clearly, from the reliability standpoint it is advantageous to introduce redundancy. However, the energy consumption of a redundant system is slightly higher than of the nonredundant system, in addition to an increase in area. The energy consumption can be lowered by completely turning off the redundant cores, instead of using sleep states. This has to be balanced by the performance cost of activating the redundant part once a failure occurs.

VII. CONCLUSION

In this work, we show that a fully functional and highly reliable core may fail to meet the lifetime requirement once power management is enabled due to thermal cycle failure mechanism. To overcome such problems, we integrated optimization of power management with a reliability constraint and developed a simulator to be used for analysis of power-reliability tradeoffs in SoCs. We have shown that with our methodology we can obtain large system power savings while meeting both performance and reliability constraints. As technology scales down, limitations set by thermal cycling are going to be an even more important factor in system design for reliability and power. Thus, joint optimization of reliability, power consumption, and performance is critical.

We plan a number of extensions to this work. First, we would like to integrate our system level simulator with a more detailed model of on-chip thermal behavior, such as HotSpot [3] at the

core level, to enable fast evaluation of power and reliability of large SoCs. Next, we plan to extend the optimizer to enable more sophisticated management policies that combine DVS, DPM, and nonstationary workloads. Additionally, we would like to study the overhead of migrating computation to other available cores since die area is limited and thus often no spare cores can be added.

REFERENCES

[1] Texas Instruments, Inc., Austin, TX, "OMAP 2420 Platform," (2007). [Online]. Available: www.ti.com

[2] S. Gunther, F. Binns, Carmean, and J. Hall, "Managing the impact of increasing microprocessor power consumption," *Intel Technol. J.*, pp. 33–45, May 2001.

[3] K. Skadron *et al.*, "Temperature-aware microarchitecture," in *Proc. Int. Symp. Comput. Arch.*, 2003, pp. 1–12.

[4] L. Shang, L. Peh, A. Kumar, and N. Jha, "Thermal modeling, characterization and management of on-chip networks," in *Proc. MICRO-37*, 2004, pp. 123–136.

[5] J. Srinivasan and S. Adve, "Predictive dynamic thermal management for multimedia applications," in *Proc. ICS*, 2003, pp. 225–228.

[6] J. Srinivasan, S. Adve, P. Bose, and J. Rivers, "Exploiting structural duplication for lifetime reliability enhancement," in *Proc. ISCA*, 2005, pp. 321–331.

[7] T. Simunic, K. Mihic, and G. De Micheli, "Reliability and power management of integrated systems," in *Proc. DSD*, 2004, pp. 12–19.

[8] Intel, Santa Clara, CA, "Intel PXA270," 2005 [Online]. Available: www.intel.com,

[9] P. Guerrier and A. Greiner, "A generic architecture for on-chip packet switched interconnections," in *Proc. DATE*, 2000, pp. 250–256.

[10] S. Kumar *et al.*, "A NOC architecture and design methodology," in *Proc. ISVLSI*, 2002, pp. 105–112.

[11] E. Rijpkema *et al.*, "Trade-offs in the design of a router with both guaranteed and best-effort services for NOCs," in *Proc. DATE*, 2003, pp. 337–344.

[12] A. Jantsch and H. Tenhunen, *Networks on Chip*. Norwell, MA: Kluwer, 2003.

[13] T. Ye, L. Benini, and G. De Micheli, "Analysis of power consumption on switch fabrics in network routers," in *Proc. DAC*, 2002, pp. 254–260.

[14] M. D. Beaudry, "Performance-related reliability measures for computing systems," *IEEE Trans. Comput.*, vol. C-27, no. 6, pp. 540–547, Jun. 1978.

[15] P. Shivakumar *et al.*, "Exploiting microarchitectural redundancy for defect tolerance," in *Proc. 21st Int. Conf. Comput. Design*, 2003, pp. 358–364.

[16] T. Simunic and S. Boyd, "Managing power consumption in networks on chips," in *Proc. Design, Autom. Test Eur.*, 2002, pp. 110–116.

[17] Q. Qiu, Q. Wu, and M. Pedram, "Dynamic power management in a mobile multimedia system with guaranteed quality-of-service," in *Proc. Design Autom. Conf.*, 2001, pp. 701–707.

[18] T. Simunic, L. Benini, P. Glynn, and G. De Micheli, "Event-driven power management," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 7, pp. 840–857, Jul. 2001.

[19] E. Rotenberg, "AR/SMT: A microarchitectural approach to fault tolerance in microprocessors," in *Proc. Int. Symp. Fault Tolerant Comput.*, 1998, pp. 111–118.

[20] J. Srinivasan, S. V. Adve, P. Bose, J. Rivers, and C. K. Hu, "RAMP: A model for reliability aware microprocessor design," IBM, Poughkeepsie, NY, 2003.

[21] K. Skadron, T. Abdelzaher, and M. R. Stan, "Control-theoretic techniques and thermal-RC modeling for accurate localized dynamic thermal managementproceedings," in *Proc. 8th Int. Symp. High-Perform. Comput. Arch. (HPCA)*, 2002, pp. 34–44.

[22] International Sematech Technology, Austin, TX, "Semiconductor device reliability failure models," 00053955A-XFR, 2000.

[23] H. V. Nguyen, "Multilevel interconnect reliability on the effects of electro-thremomechanical stresses," Ph.D. dissertation, Dept. Mech. Eng., Univ. Twente, Twente, The Netherlands, 2004.

[24] M. Huang and Z. Suo, "Thin film cracking and ratcheting caused by temperature cycling," *J. Mater. Res.*, vol. 15, no. 6, pp. 1239–1243, Jun. 2000.

[25] R. Degraeve and J. L. Ogier *et al.*, "A new model for the field dependence of intrinsic and extrinsic time-dependent dielectric breakdown," *IEEE Trans. Elect. Devices*, vol. 45, no. 2, pp. 472–481, Feb. 1998.

[26] E. E. Lewis, *Introduction to Reliability Engineering*. New York: Wiley, 1996.

[27] Texas Instruments, Inc., Austin, TX, "TMS320C6211, TMS320C6211B, Fixed-point digital signal processors," 2002.

[28] Philips Semiconductors, San Jose, CA, "SAF7113H datasheet," 2004.

[29] Analog Devices, San Jose, CA, "SST-melody-DAP audio processor," 2002.

[30] Texas Instruments, Austin, TX, "MSP430x11x2, MSP430x12x2 mixed signal microcontroller," 2004.

[31] Rambus, Sunnyvale, CA, "RDRAM 512 Mb," 2003.

[32] A. Maheshwari, W. Burleson, and R. Tessier, "Trading off reliability and power-consumption in ultra-low power systems," in *Proc. ISQED 2002*, 2002, pp. 432–443.

[33] P. Stanley-Marbell and D. Marculescu, "Dynamic fault-tolerance and metrics for battery powered, failure-prone systems," in *Proc. ICCAD*, 2003, pp. 435–441.

[34] N. Shanbhag, "Reliable and efficient system-on-chip design," *IEEE Comput.*, vol. 37, no. 3, pp. 42–50, Mar. 2004.

[35] D. Ernst, N. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "Razor: A Low-power pipeline based on circuit-level timing speculation," in *Proc. MICRO-36*, 2003, pp. 536–547.

[36] R. A. Sahner, K. S. Trivedi, and A. Puliafito, *Performance and Reliability Analysis of Computer Systems: An Example-Based Approach Using the SHARPE Software Package*. Norwell, MA: Kluwer, 1996.

Tajana Simunic Rosing (M'76–SM'81–F'87) received the M.S. degree in electrical engineering from the University of Arizona, Tucson, in 1993, where her M.S. thesis topic was high-speed interconnect and driver-receiver circuit design. She also received the M.S. degree in engineering management and the Ph.D. degree from Stanford University, Stanford, CA, in 2001. Her Ph.D. dissertation was on dynamic management of power consumption.

She is currently an Assistant Professor in the Computer Science Department, University of California (UCSD), San Diego, La Jolla. Her research interests include low-power system design, embedded systems, and wireless system design. Prior to this, she was a full time Researcher at HP Laboratories, Palo Alto, CA, while working part-time at Stanford University, Stanford, CA, where she has been involved with leading research of a number of graduate students and has taught graduate level classes. Prior to pursuing the Ph.D. degree, she worked as a Senior Design Engineer at Altera Corporation, San Jose, CA. She has served at a number of Technical Paper Committees, and has been an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS.

Kresimir Mihic (M'76–SM'81–F'87) is currently pursuing the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA.

From 2005 to 2007, he was a Product Engineer at Magma Design Automation, Santa Clara, CA. Between 2000 and 2005, he was a Process Engineer at Cypress Semiconductor, San Jose, CA. His research interests include SoC reliability and dynamic power management.

Giovanni De Micheli (F'94) received the Nuclear Engineer degree from Politecnico di Milano, Milano, Italy, in 1979, and the M.S. and the Ph.D. degrees in electrical engineering and computer science from the University of California at Berkeley, in 1980 and 1983.

He is a Professor and a Director of the Integrated Systems Centre, EPF Lausanne, Switzerland, and the President of the Scientific Committee of CSEM, Neuchatel, Switzerland. Previously, he was a Professor of Electrical Engineering at Stanford University, Stanford, CA. His research interests include several aspects of design technologies for integrated circuits and systems, such as synthesis, hw/sw codesign and low-power design, as well as systems on heterogeneous platforms including electrical, optical, micromechanical, and biological components. He is the author of *Synthesis and Optimization of Digital Circuits* (McGraw-Hill, 1994), co-author and/or co-editor of six other books, and of over 300 technical articles. He is, or has been, a member of the technical advisory board of several companies, including Magma Design Automation, Coware, Aplus Design Technologies, Ambit Design Systems, and STMicroelectronics.

Prof. De Micheli was a recipient of the 2003 IEEE Emanuel Piore Award for contributions to computer-aided synthesis of digital systems. He received the Golden Jubilee Medal for outstanding contributions to the IEEE CAS Society in 2000, the 1987 D. Pederson Award for the best paper on the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, two Best Paper Awards at the Design Automation Conference, in 1983 and in 1993, and a Best Paper Award at the DATE Conference in 2005. He has been serving IEEE in several capacities, namely: Division 1 Director (2008–9), cofounder and President Elect of the IEEE Council on CEDA (2005–7), President of the IEEE CAS Society (2003), Editor in Chief of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS (1987–2001). He was Program Chair of the pHealth and VLSI SOC conferences in 2006. He was the Program Chair and General Chair of the Design Automation Conference (DAC) in 1996, 1997, and 2000, respectively. He was the Program and General Chair of the International Conference on Computer Design (ICCD) in 1988 and 1989, respectively. He is a founding member of the ALaRI Institute at Universita' della Svizzera Italiana (USI), Lugano, Switzerland, where he is currently a Scientific Counselor. He is a Fellow of Association of Computing Machinery (ACM).