Uncertainty-Aware Dynamic Power Management in Partially Observable Domains

Hwisung Jung, *Student member*, *IEEE*, Massoud Pedram, *Fellow, IEEE*

***Abstract* - This paper tackles the problem of dynamic power management (DPM) in nanoscale CMOS design technologies that are typically affected by increasing levels of process and temperature variations and fluctuations due to the randomness in the behavior of silicon structure. This uncertainty undermines the accuracy and effectiveness of traditional DPM approaches. This paper presents a stochastic framework to improve the accuracy of decision making during dynamic power management, while considering manufacturing process and/or environment induced uncertainties. More precisely, variability and uncertainty at the system level are captured by a partially observable semi-Markov decision process with interval-based definition of states while the policy optimization problem is formulated as a mathematical program based on this model. Experimental results with a RISC processor in 65nm technology demonstrate the effectiveness of the technique and show that the proposed uncertainty-aware power management technique ensures system-wide energy savings under statistical circuit parameter variations.**

***Index Terms* — Dynamic power management, stochastic control, POMDP, uncertainty**

I. Introduction

IC designers are seeking high-performance and reliable electronic circuits and systems. As we start to design with the nanometer process technology nodes, IC design methodologies based on zero guard band or design margins are becoming unsatisfactory due in part to the increasing levels of process variations (i.e., oxide thickness and channel length variations) and random circuit parameter fluctuations (i.e., dopant fluctuations). These variations, especially within-chip variations, pose a major challenge to the design of low-power and high-performance circuits and systems. Uncertainties which arise either from environmental changes (i.e., temperature gradients or voltage droops) or are application (workload) dependent also give rise to additional complications when trying to optimize the energy consumption and performance of a state-of-the-art electronic system.

Variability means that different applications or situations produce different numerical values for a quantity. Specifying an exact value for a quantity may be difficult because the value depends on something else. For example, the amount of power consumed by a system depends upon the chip temperature and the workload mix of the applications running on that system. The existence of variability in a population implies that a single action or strategy may not emerge as optimal for each of the individuals, and consequently any decision made will go too far for some and not far enough for others. Uncertainty arises due to lack of knowledge regarding the true value of a quantity for a given member of a population. When making observations of past events or speculating about the future, imperfect knowledge is the rule rather than the exception. For example, the workload attributed to a particular functional unit is not directly recorded very often. Rather, this workload is usually recorded at fixed intervals and then extrapolated based upon historical data and trend lines. Note that direct measurements also have some margin of error. When estimating numerical values expected for workload at some future date, the exact outcome is rarely known in advance. More generally, sources of uncertainty include: uncertainty of data and parameters of models, uncertainty about choice of models, and uncertainty about the future. Uncertainty implies that we might make a non-optimal choice because we may expect one outcome but something quite different might actually occur.

Uncertainty and variability typically require different treatments. For example, if we seek to develop a dynamic power management (DPM) methodology that can optimize energy dissipation and performance of different instances of the same system design, it is important that we separate sources of uncertainty (which affect the underlying system state) from sources of variability (which affect the cost rate for being in any state) so that we can properly model the system, and hence perform policy optimization accurately and reliably. This statement is true although variability contributes to uncertainty, and the amount of variability is generally itself an uncertain parameter.

Most of the previous work on variability has focused on the variability modeling and analysis at the lower levels of design abstraction [1]-[3], and the corresponding circuit-level or physical design optimization techniques [4]-[7]. It is only recently that people have started paying attention to the effects of variability on optimization processes and tradeoffs as we go up in the design abstraction hierarchy [8]-[10]. It is prudent to account for various sources of variability even earlier in the design process and for the whole design when developing resource management and power control strategies. It is important to note that some variations at the higher levels of design abstraction are translated into uncertainty because the underlying RT-level/physical realization is not available. At the same time, measurements made about the current state of the system tend to be imperfect, which in turn gives rise to

Manuscript received July 27, 2007; revised January 18, 2008. This work was supported in part by the National Science Foundation under grant no. 0509564.

The authors are with the Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90007 USA (e-mail: hwijung@usc.edu; pedram@usc.edu).

uncertainty about the system state. Improving the accuracy and robustness of decision making by modeling and assessing the variability and uncertainty is an important step in guaranteeing the quality of system-level resource management algorithms, including DPM.

This papers tackles the problem of system-level dynamic power management (DPM) in systems which are manufactured in nanoscale CMOS technologies and are operated under widely varying conditions over the lifetime of the system. Such systems are greatly affected by increasing levels of process variations typically materializing as random or systematic sources of variability in device and interconnect characteristics, and widely varying workloads and temperature fluctuations usually appearing as sources of uncertainty. At the system level this variability and uncertainty is beginning to undermine the effectiveness of traditional DPM approaches. It is thus critically important that we develop the mathematical basis and practical applications of a variability-aware, uncertainty-reducing DPM approach with the following unique feature: Utilization of a stochastic modeling framework based on the theory of partially observable semi-Markovian Decision Model (POSMDP) [11], which can efficiently cope with uncertainty. We also present uncertainty-aware offline/online dynamic power management techniques to illustrate the effectiveness of the uncertainty management framework. A preliminary version of this research appeared in [12].

The remainder of this paper is organized as follows. The related work is discussed in section II. In section III, the preliminaries of the paper are presented. The details of the stochastic uncertainty management framework are given in section IV. The policy representation for the proposed framework is described in section V. Section VI presents uncertainty-aware dynamic power management techniques. Experimental results and conclusion are given in section VII and section VIII.

II. Related Work

Increasing attention has been given to the problem of reducing variability in the circuit design parameters. In the following, we provide a brief overview of these works. The work presented in [13] studies the impact of leakage reduction techniques on the delay uncertainty. By emphasizing that the leakage is critically dependent on the operating temperature and power supply, the authors in [14] present a full chip leakage estimation technique which accurately accounts for power supply and temperature variations. In [15], the authors discuss process, voltage, and temperature variations and their impacts on circuit and micro-architectures beyond the 90nm technology node. Probabilistic models are introduced in [16] to account for the impact of threshold voltage variations on the leakage power. These models are subsequently employed to minimize the leakage power, while satisfying a given performance requirement. Reference [17] presents a technique to optimize supply and threshold voltage in high-performance circuits. The authors show that interactions between supply voltage, frequency, power, and temperature significantly impact the energy-delay-product of a target design. None of these works has considered the effect of variability on system-level dynamic power management.

A lot of research has been devoted to optimizing DPM policies, resulting in both heuristics and stochastic approaches. While the heuristic approaches are easy to implement, they do not provide and power/performance assurances. In contrast, the stochastic approaches guarantee optimality under performance constraints although they are more complex to implement [18]. To overcome the limitations of heuristic "time-out"-based power management techniques, an approach based on discrete-time Markovian decision processes (DTMDP) was proposed in [19]. This approach outperforms the previous heuristic techniques because of its solid theoretical framework for system modeling and policy optimization. We introduced a power management approach based on continuous-time Markovian decision processes (CTMDP) in [20]. The policy change framework this model is asynchronous and thus more suitable for implementation as part of a real-time operating system environment.

Reference [21] also improved on the modeling technique of [19] by using time-indexed semi-Markovian decision processes (SMDP). An SMDP is a stochastic process where the next state depends on the current state and how long the current state has been active. A non-stationary process based power management technique is introduced in [22], where the workload requests are modeled as a Markov-modulated stochastic process. In [23] we introduced a hierarchical power management architecture which aims to facilitate power-awareness in a system with multiple components, each having a built-in local power manager. The proposed architecture divides the power management function into two layers: system-level and component-level. The system-level power management is formulated as a concurrent service request flow regulation and application scheduling problem.

The above-mentioned stochastic power management techniques enjoy desirable features of flexibility, global optimality, and mathematical robustness. In general, however, these models are somewhat limited in their reach and applicability because they assume that various variables of the system are directly observable and thus are deterministic and that there are no sources of variability in the design.

These works do not make a distinction between uncertainty and variability. They typically assume that all system parameters are either precisely known scalar values or are random variables with known mean and an irreducible variance. They thus miss out on the importance of identifying sources of uncertainty in the distributions that can be reduced or even eliminated through measurements/observations and accurate modeling.

In summary, to the best of our knowledge, there has been no reported work on DPM with stochastic modeling and appropriate treatment of uncertainty and variability. This is the aim of the present proposal. An integrated dynamic power management framework makes it possible to consider the stochastic behavior of power dissipation and performance of a system and provide the computational tractability of the randomness to treat the many sources of variability and utilize direct observations and models to control sources of uncertainty, bringing the underlying variability and

randomness effects to the forefront of power management policy optimization.

III. PRELIMINARIES

As nanoscale VLSI circuits are becoming sensitive to the rising levels of variability in process and design parameters, guaranteeing the quality of system-level performance optimization techniques is becoming of great concern. Within-chip variations are typically passed into the delay budget of each circuit [26]. However, the worst-case behavior of the circuit (e.g., critical path delay) does not always correspond to the combination of worst-case points of individual parameters, e.g., load capacitance, intrinsic delay, and slew rate. Furthermore, a lot of Silicon performance is left untapped under the worst-case assumption. IC designers can no longer afford to lose performance due to unacceptable levels of inaccuracy in their estimation/modeling techniques [27]. Thus, it is important to do rigorous modeling of variability early in the design cycle.

A. Effect of PVT Variations on the Performance State

Although performance analysis tools provide reliable bounds on the delay of circuits, they cannot properly account for the variability inherent in the semiconductor process. For example, Fig. 1 illustrates the effect of variations on propagation delays of logic gates (e.g., 2-input NAND gate driving FO4 load) as calculated by 2-D lookup tables, which are used in conventional performance analysis tools (e.g., PrimeTime [28]) under the worst corner case (125°C, 1.08V for V_{dd}) of 65nm CMOS technology. Every point in the table represents characterized spice delay for the logic gate for a particular input transition time and output capacitance pair.

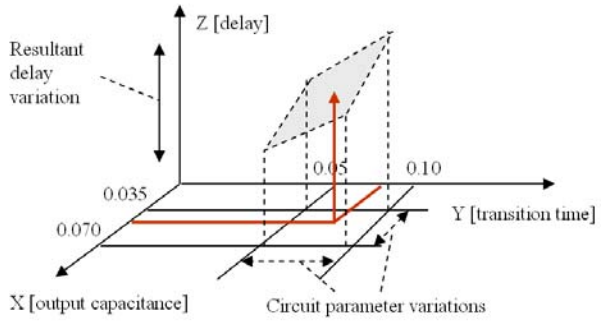


Fig. 1. Effect of process variations on circuit delay.

Obviously, not all possible input transitions and output capacitance values for a given cell can be characterized. In this figure, the closet four characterized points in the table are interpolated to provide a desired output delay value. Thus, although these analysis tools can provide estimates of performance parameters at design time, they cannot guarantee that the expected performance prediction is accurate in manufactured designs.

We reiterate the obvious fact that voltage (V) and temperature (T) variations are dynamic, i.e., they occur during the circuit operation, whereas process (P) variations are static and get introduced during the manufacturing. The strong impact of PVT variations on performance of a VLSI circuit renders the traditional optimization techniques ineffective. This phenomenon has resulted in a move toward stochastic optimization strategies, i.e., techniques that treat design parameters as random values whose values are described by probability distribution functions. As an example, Fig. 2 shows a number of leakage power-delay tradeoff curves for a RISC processor that we designed in 65nm CMOS technology, where the delay represents execution time of a target task by the processor. The curves are obtained by running the Synopsys Power Compiler [29] for three different process conditions. Note that SS, TT, and FF conditions represent [1.08Vdd, 125°C], [1.20Vdd, 25°C], and [1.29Vdd, -40°C], respectively.

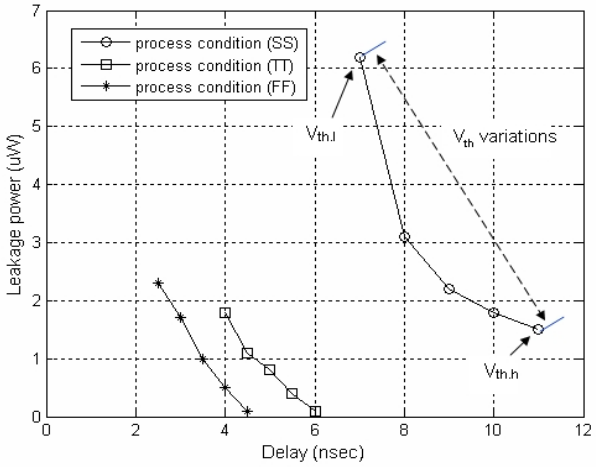


Fig. 2. Power-delay curves of a RISC processor corresponding to different process corners in a 65nm CMOS process technology node (S, T and F stand for Slow, Typical and Fast nMOS or pMOS transistors).

The task of computing power management policies that cope with uncertainty and non-determinism require the construction of a stochastic framework with which one can predict the effect of various actions on the performance state of the system. A stochastic approach to system-level performance modeling and optimization (e.g., one based on the Markovian decision process model) enables us to apply mathematical optimization techniques to derive optimal policies for DPM.

B. Temperature Calculation

The major source of heat generation in a die is the power dissipation of transistors whose active regions are implemented in the substrate [30]. Some amount of power dissipation also results from Joule heating (or self-heating) caused by the flow of current in the interconnections. This effect is ignored in our simulations.

Temperature of a VLSI chip can be calculated as follows:

$$T_{chip} = T_J - R_\theta \cdot \left(\frac{P_{total}}{A} \right) \quad (1)$$

where T_{chip} is the temperature of the case top, T_J is the junction temperature, R_θ is the equivalent junction-to-case thermal resistance of the substrate (Si) layer plus the package (cm^2 °C / W), P_{total} is the total power consumption (W), and A is the chip area (cm^2). In this paper, it is assumed that power density can

serve as a proxy for temperature variations although a change in instantaneous power dissipation does not give rise to an immediate temperature change due to a low-pass filtering effect in translating power variations into temperature variations [31].

IV. Stochastic Decision Making Framework

In this section, we first present the idea of using a stochastic model for dealing with the uncertainty in observations made by a power manager, and then introduce a theoretical framework for constructing the model of power manager operating in such an uncertain environment.

A. Partially Observable Environments

Generally speaking, at specific instances in time called decision epochs, a power manager observes some characteristic of the system, estimates the system performance state (e.g., its execution delay and power dissipation) on the basis of this observation, and issues a command (i.e., action) to force a state transition according to a power management policy that maximizes (or minimizes) a user-specified reward (or cost) function. The concept is that the actual state of the system, which is not directly observable, is estimated by observing some other system characteristic.

A Markov decision process (MDP) model facilitates reasoning in domains where actions change the system states and where a reward (or cost) is utilized to optimize the system performance. The simple MDP is directly observable in the sense that its execution hinges on the assumption that the current system state can be determined without any errors and that the reward (cost) of an action can be calculated exactly. In partially observable environments, where performance states of the system cannot be identified exactly, observations made by a power manager about the state of the system are indirect and may even be noisy, and therefore, they only provide incomplete information. A naive strategy for dealing with this uncertainty is to ignore the problem altogether, that is, to treat the observations as if they provide accurate and complete information about the actual state of the system and act on them. This strategy can result in undesirable decisions based on erroneous readings of the current and next states of the system.

A more sophisticated strategy resorts to stochastic modeling and decision making. One way to deal with uncertainty under a wide range of operating conditions and environments is to rely on the history of previous actions and observations to disambiguate the current state. For example, we can adopt a hidden Markov model (HMM), where the state is not directly observable but variables influenced by the state are observable, to learn a model of the environment, including the hidden states [12]. Note that in an HMM each state has a probability distribution over the possible actions, resulting in the fact that the sequence of actions generated by the HMM gives some information about the sequence of states. Thus, a power manager in the HMM reasons about the state of the system indirectly through the observed variables, which captures complex system dynamics which are not completely observable.

B. Sequential Decision Making under Uncertainty

The decision making in a partially observable environment is achieved by combining aspects of HMMs and MDPs. Specifically, we start with a semi-Markov decision process (SMDP), a generalization of MDPs, to model the decision making strategy, and then combine it with a HMM to consider the uncertainty in parameter observation. We call this combination a partially observable semi-Markov decision process (POSMDP) model. Recall that inter-arrival times of requests in the SMDP model follow an arbitrary distribution, which is a more realistic assumption than an exponential distribution used in the conventional MDP model.

Definition 1: Partially Observable Semi-Markov Decision Process. A POSMDP is a tuple (*S, A, O, T, Z, k*) such that

1) S is a finite set of states,
2) A is a finite set of actions,
3) O is a finite set of observations,
4) T is a transition probability function,
5) Z is an observation function, and
6) k is a cost function,

The state space S comprises of a finite set of state, where $s \in S$ can be defined as performance state of the system. The action space A consists of a finite set of action $a \in A$, e.g., dynamic voltage and frequency scaling (DVFS) values which control the performance state of the system. The observation space O contains a finite set of observation $o \in O$, e.g., on-chip temperature measurement. The state transition probability function, $T(s^{t+1}, a^t, s^t)$ [1], determines the probability of a transition from a state s^t to another state s^{t+1} after executing action a^t, i.e., the system transits to the state s^{t+1} at time t+1 with probability $Pr(s^{t+1} \mid s^t, a^t) = T(s^{t+1}, a^t, s^t)$. An observation function, $Z(o^{t+1}, s^{t+1}, a^t)$, which captures the relationship between the actual state and the observation, is defined as the probability of making observation o^{t+1} after taking action a^t that has landed the system in state s^{t+1}, i.e., state s^{t+1} generates observation o^{t+1} at time t+1 with probability $Pr(o^{t+1} \mid s^{t+1}, a^t) = Z(o^{t+1}, s^{t+1}, a^t)$. We consider a cost function that assigns a real-valued number to each state and action pair whereby an immediate cost, $k(s, a)$, is incurred when action a is chosen in state s. A solution to a POMDP is a policy (a procedure for selecting an action in every state) that minimizes some measure of aggregate cost, called objective function.

The objective function maps infinite sequences of costs to a single value, which is typically infinite. How do we compare policies of infinite cost? We have three options: (i) Set a finite horizon and simply sum the cost, (ii) Discount to prefer earlier costs, and (iii) Use the average cost in the limit. A value function, V_π, represents the expected objective function value obtained following policy π starting from each state in S. Value functions partially order the policies, but at least one optimal policy exists, and all optimal policies have the same value function, V^*. Bellman equations [36] relate the value function to itself via the problem dynamics. For the discounted objective function, with a discount rate of $0 \le \gamma < 1$, they are:

[1] In this paper, subscripts denote state information whereas superscripts denote time stamp.

$$\forall s: \quad V_\pi(s) = C(s,a) + \gamma \sum_{s' \in S} T(s', a, s) V_\pi(s') \quad \text{with} \quad a = \pi(s)$$

$$V^*(s) = \min_{a \in A} \left(C(s,a) + \gamma \sum_{s' \in S} T(s', a, s) V^*(s') \right)$$

Instead of making decisions based on the current perceived state of the system, the POSMDP maintains a *belief*, i.e., a probability distribution over the possible (nominal) states of the system, and makes decisions based on its current belief. The *belief state* at time t is a $|S|\times 1$ vector of probabilities defined as: $b^t := [b^t(s)]$, $\forall s \in S$, where $b^t(s)$ is the posterior probability distribution of state s at time t. Note that $\Sigma_{s \in S} b^t(s) = 1$ [32]. Based on the belief state, an action a^t is chosen from a set of available actions. A *policy* is defined as a sequence of mappings from the belief states to actions $\pi = \{\pi^t\}$.

In this paper, we consider a design scenario where actions incur a cost (i.e., energy dissipation), and the power manager's goal is to devise a policy that minimizes the total expected energy dissipation. Fig. 3 illustrates the basic structure of a POSMDP-based power manager. The proposed power manager interacts with an uncertain environment and statistically variable state variables and tries to minimize the system cost over time by choosing appropriate actions. The frequency-voltage level assignment actions issued by the power manager change the performance state (power dissipation and speed) of the system and lead to quantifiable rewards/penalties.

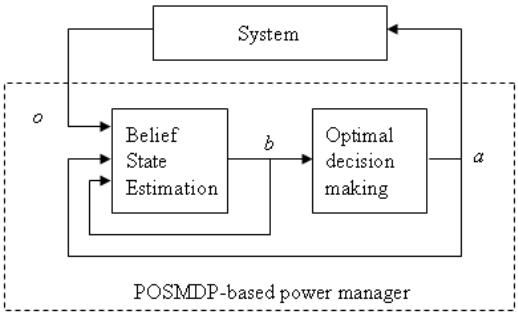


Fig. 3. Structure of a POSMDP-based power manager.

In our formulation of the decision-making strategy, we define state $s \in S$ as the dissipated power level and largest stage delay of the circuit. Furthermore, we use an *observation*, i.e., a temperature measurement to help determine the system state.[2] The power manager consists of two functional components. The first component is the belief state estimation block which computes the system's belief state, while the second component is the decision making block which assigns optimal actions to the system based on a value-iteration policy optimization algorithm.

Consider a three-state DPM problem as an example. A graphical representation of the belief state and its evolution is provided in Fig. 4. In this figure, starting in some current belief state, we show the next belief states depending on the action.

[2] Note that other runtime observations/measurements may be used to help with this determination, for example, it is possible to replicate the worst-case execution path of a circuit and monitor its actual delay at runtime. This, however, has hardware overhead.

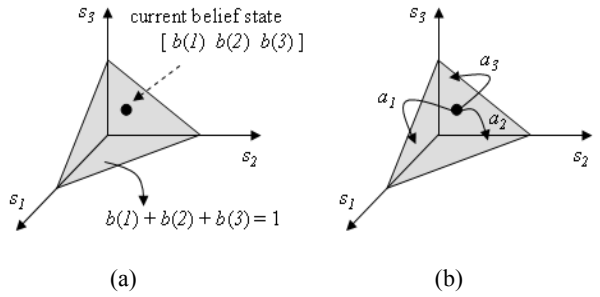


Fig. 4. A graphical representation of the belief state: (a) current belief (b) its one-step evolution for three different actions.

C. POSMDP Framework for Dynamic Power Management

The rationale for developing a POSMDP framework for dynamic power management is depicted in Fig. 5. First, since the performance state of a system cannot be directly determined by the PM, it uses temperature readings to help estimate the current system state in the form of a belief state. We assume that the chip temperature at time t is one of three observations: o_1, o_2, and o_3 corresponding to different, but well-specified, temperature ranges. The system state at time t is defined as a combination of delay (e.g., d_1, d_2, or d_3, where $d_1 < d_2 < d_3$) and power dissipation (e.g., p_1, p_2, or p_3, where $p_1 < p_2 < p_3$) values. Starting from system state $s^t(d_2, p_3)$ at time t, the power manager issues an action, $a^t = (V_{dd1}, freq_2)$, and as a result, the system is expected to move into a new state $s^{t+1}(d_3, p_2)$ at time $t+\varepsilon$. Let's assume that, due to variations, the resulting system state is actually $s^{t+1}(d_3, p_3)$. Since state s^{t+1} is not directly observable, the PM must rely on observation o^{t+1} at time $t+1$ to estimate the state that it is in.

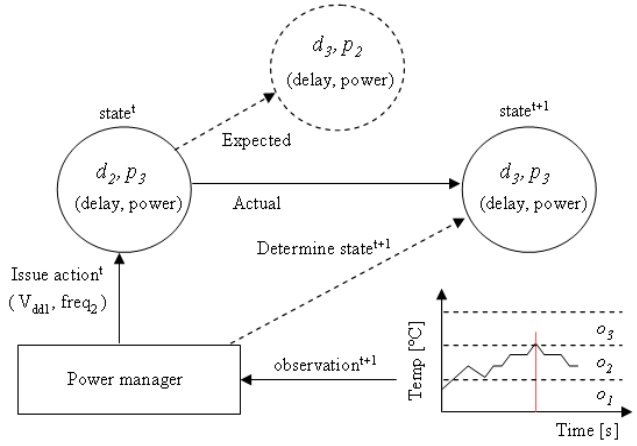


Fig. 5. State estimation and state transition in the POSMDP-based DPM.

Fig. 6 illustrates yet another uncertainty effect. More precisely, the figure shows three scenarios where starting from current state $s^t(d_2, p_2)$ with an action a_2, e.g., [1.20V / 650MHz] issued at time t, the next system state may be any one of three possible states at time $t+1$, that is, the power manager cannot know for certain which next state will occur, although it will have some information from the observation, o^{t+1}. For example, in case (a), the system remains in the same active state after a_2 is chosen, resulting in the same performance (i.e., $s^{t+1}(d_2, p_2)$). That is why decisions will be made based on the probability

distribution vector of the belief state, b^{t+1}.

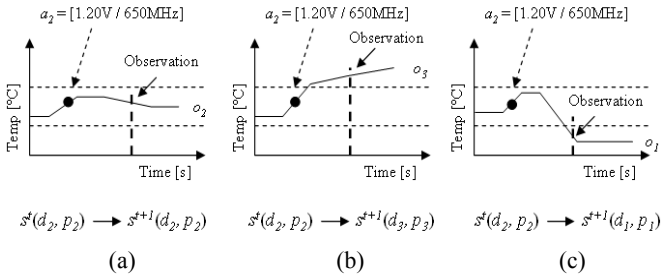


Fig. 6. Example of three possible observations at time t+1 from which the belief state is calculated.

V. Policy Representation in POSMDP

We provide a policy representation of the proposed power management framework by presenting a belief-state SMDP, and derive the optimal power management policy.

A. Conversion to Belief-state SMDP

In partially observable environment, a power manager can make decisions based on the observed system state history H since the underlying performance state of the system cannot be fully observed. Note that the system history H is a sequence of state and action pair such as $<s^0, a^0>, <s^1, a^1>, \ldots, <s^t, a^t>$. Thus, the power manager's behavior is determined by its policy, which is a mapping from the set of observable history H to the action set A, where the power manager can only base its decisions on the history of its actions and states. This means that complete history of system states is relevant to predicting the future state of the system, which makes this decision making process a *non-Markovian* process [11]. Fortunately, the power management problem may also be formulated as a Markovian process-based optimization problem as proved in [33]. More precisely, we can convert the above-mentioned non-Markovian process into a *Markovian* process when formulating the power management problem as follows. To achieve the Markovian property, we make use of the belief state, b. It has been shown that the belief state is sufficient in the sense that it completely captures the power manager's knowledge about the current state and past history [34].

Given belief state b^t and an action a^t resulting in observation o^{t+1}, we can compute the successor belief state b^{t+1} as follows:

$$b^{t+1}(s) = Pr(s \mid o^{t+1}, a^t, b^t) = \frac{Z(o^{t+1}, s, a^t) \cdot \sum_{s'} b^t(s') \cdot T(s, a^t, s')}{\sum_{s''} \left(Z(o^{t+1}, s'', a^t) \sum_{s'} \left(b^t(s') \cdot T(s'', a^t, s') \right) \right)} \quad (2)$$

In (2), the numerator consists of the product of the probability that observation o^{t+1} is made in state s after action a^t is taken, and the probability that starting from belief state b^t, we end up in state s under action a^t. The denominator denotes the probability of perceiving o^{t+} given action a^t and belief state b^t. Note that the $|S|$-dimensional belief state is continuous.

The belief state transition function, $T_b(b^{t+1}, a^t, b^t)$, which provides the probability of a transition from current belief state b^t to next belief state b^{t+1} after executing action a^t, is given by:

$$T_b(b^{t+1}, a^t, b^t) = Pr(b^{t+1} \mid b^t, a^t) = \sum_o Pr(b^{t+1} \mid a^t, b^t, o) \cdot Pr(o \mid a^t, b^t) \quad (3)$$

The probability of perceiving o, given action a^t and belief state b^t, is given by summing over all the actual states that may be reached, i.e.,

$$Pr(o \mid a^t, b^t) = \sum_{s'} Z(o, s', a^t) \cdot \sum_{s} T(s', a^t, s) b^t(s)$$

As stated earlier, the key result is that if we maintain and update the belief state and transition probabilities according to (2) and (3), then the belief state will give us with just as much information as the entire action-observation history. This shows that the optimal POSMDP solution is Markovian over the belief space. Hence, by using the belief space B, we can convert the original POSMDP into a completely observable, regular (albeit continuous state space) semi-Markov decision process (SMDP), the so-called belief state SMDP, defined as follows.

Definition 2: Belief state SMDP is a tuple (B, A, T_b, C_b) such that

1) B is the belief space,
2) A is the set of actions,
3) T_b is the belief state transition function, and
4) C_b is the cost function,

where the updated belief state after action a can be calculated from the previous belief state from (2). The belief state transition function is given by (3). We also need a model for system cost based on belief states:

$$C_b(b^t, a^t) = \sum_s b^t(s) k(s, a^t) \quad (4)$$

which denotes the immediate cost incurred by action a^t issued in current state b^t. Here, $k(s,a^t)$ denotes the immediate cost of action a^t in state s.

We have thus transformed the problem formulation based on the POSMDP model to one based on belief-state SMDP model. The optimal policy, $\pi^*(b)$ of the belief-state SMDP representation is also optimal for the physical-state POSMDP representation. Notice that the belief-state SMDP model is deterministic and fully observable because it already takes into account the uncertainty.

B. Policy Representation

Finding an optimal power management policy requires a decision-making strategy which maps the belief states to actions. In this paper, we develop a policy generation technique by using well-known dynamic programming method, which in turn relies on principles of overlapping subproblems, optimal substructures, and memorization. We speak of the minimum value of a system state as the expected infinite discounted sum of cost that the system will accrue if it starts in that state and executes the optimal policy [35]. The goal is to minimize some cumulative function of the costs, typically the infinite-horizon sum under a discounting factor γ (usually just under 1). This would look like:

$$V^*(\pi) = \min_{\pi} E\left(\sum_{t=0}^{\infty} \gamma^t C_b(t)\right) = \min_{\pi}\left(\sum_{t=0}^{\infty}\sum_{b^t,a^t} \gamma^t T_b(b^{t+1}, a^t, b^t) C_b(b^t, a^t)\right) \quad (5)$$

where γ is a discount factor, $0 \le \gamma < 1$, the exponent t denotes the duration of time that the system spends in the belief state b before an action a causes a transition to another state b', and $C_b(t)$ is the cost at time t. E(.) denotes the expectation value.

The standard family of algorithms [36] to calculate the policy requires storage for two arrays indexed by state: *value* V, which contains real values, and *policy* π which contains actions. At the end of the algorithm, π will contain the solution and $V(s_0)$ will contain the discounted sum of the costs to be accrued (on average) by following that solution. The algorithm then has the following two kinds of steps, which are repeated in some order for all the states until no further changes take place.

$$\pi(b) = \arg\min_{a}\left(\sum_{b' \in B} T_b(b', a, b)\Phi(b')\right)$$
$$V(b) = C_b(b^t, a^t) + \gamma \sum_{b' \in B} T_b(b', \pi(b), b) V(b') \quad (6)$$

In *value iteration* (cf. Fig. 7), the π array is not used; instead, the value of $\pi(s)$ is calculated whenever it is needed. Substituting the calculation of $\pi(s)$ into the calculation of $V(s)$ gives the combined step:

$$V^*(b) = \min_{a}\left(C_b(b^t, a^t) + \gamma \sum_{b' \in B} T_b(b', a, b) V^*(b')\right)$$
$$\pi^*(b) = \arg\min_{a}\left(C(b^t, a^t) + \gamma \sum_{b' \in B} T_b(b', a, b) V^*(b')\right) \quad (7)$$

The first equation asserts that the value of a state b is the expected immediate cost plus the expected discounted cost of the next state, using the best available action. Assuming that the value function V is additive, which is reasonable in our problem context since the cost is defined as the energy dissipation of the system over time, the second equation specifies the optimal policy based on the optimal value function.

Simply stated, the power manager determines the optimal action based on Eqn. (7) at each (e.g., time-based or interrupt-based) decision epoch. The task of casting the decision epochs to absolute time units is achieved by the system developer. In this paper, we consider battery operated systems that strive to conserve energy to extend the battery life.

Unfortunately, it is not obvious when to stop the value iteration algorithm. A key result bounds the performance of the current greedy policy as a function of the Bellman residual of the current cost function [37]. It states that if the maximum difference between two successive cost functions is less than ε, then the cost of the greedy policy (i.e., the policy obtained by choosing, in every state, the action that minimizes the estimated discounted cost, using the current estimate of the cost function) differs from the cost function of the optimal policy by no more than $2\varepsilon\gamma/(1-\gamma)$ at any state. This provides a stopping criterion for the algorithm.

```
1: initialize V(b) arbitrarily
2:   loop until policy good enough
3:     loop for ∀b ∈ B
4:       loop for ∀a ∈ A
5:       Q(b^t, a^t) = C_b(b^t, a^t) + γ^t Σ_{b'∈B} T_b(b', a, b) V(b')
6:       V*(b) = min_a Q(b^t, a^t)
7:       end loop
8:     end loop
9:   end loop
```

Fig. 7. The value iteration algorithm.

C. POSMDP-based DPM by Example

An example of value iteration for the POSMDP model is given next. The purpose of the example is to show how to find the best action by building value functions. We consider the POSMDP framework of a power manager with two system states, $S = \{s_1, s_2\}$, where s_1 denotes a low-power (low-performance) system state whereas s_2 corresponds to a high-power (high-performance state; two actions, $A = \{a_1, a_2\}$, where a_1 commands a low-voltage, low-frequency setting whereas a_2 commands a high-voltage, high-frequency assignment to the system; and finally two temperature observations, $O = \{o_1, o_2\}$, where o_1 corresponds to a low temperature range whereas o_2 denotes a high temperature reading.

The parameter values are given in TABLE I. We also specify the immediate values of the two actions. Let action a_1 have a value of 1.0 if it is issued in state s_1 and 0.8 in state s_2. Similarly, let action a_2 have a value of 0.4 and 1.5 in states s_1 and s_2, respectively, i.e., $k(s_1, a_1) = 1.0$, $k(s_2, a_1) = 0.8$, $k(s_1, a_2) = 0.4$, and $k(s_2, a_2) = 1.5$. (cf. Fig. 8).

TABLE I
PARAMETER VALUES FOR THE EXAMPLE PROBLEM

State	Description		Action	Description	Observation	Description
	delay (ns)	power (mW)				
s_1	$d > 8$	$p \le 22$	a_1	[1.08V / 500MHz]	o_1	[50°C ≤ temp < 65°C]
s_2	$d \le 8$	$p > 22$	a_2	[1.20V / 650MHz]	o_2	[65°C ≤ temp < 75°C]

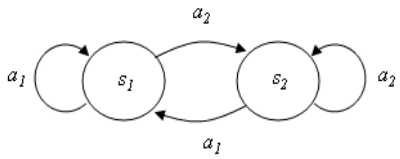
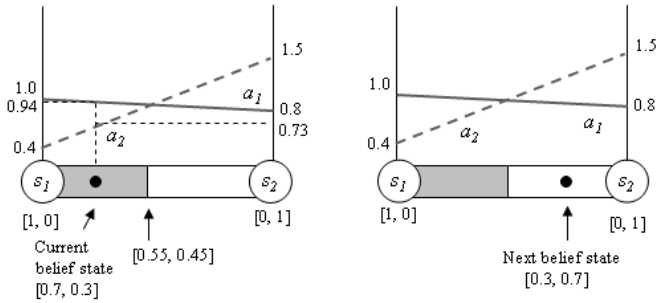


Fig. 8. State transition diagram for the example problem.

Referring to Fig. 9, the two actual system states $\{s_1, s_2\}$ are labeled by belief states [1, 0] on the left (i.e., state is s_1 with probability 1), and [0, 1] on the right (i.e., state is s_2 with probability 1). The solid line represents the value of taking action a_1, while the dashed line represents the value of taking action a_2. The actual belief state is a probability distribution over the two states, s_1 and s_2. Assuming that the initial belief state is [0.7 0.3], we will show how to construct the value function from which we determine the best action (i.e., one

with the lowest value) when we consider only a sequence of two actions from any belief state (i.e., the horizon length is 2).

The first step is to find the immediate values of choosing actions. For example, by applying (4), the immediate value of doing action a_1 in the initial belief state b is (0.7×1.0)+(0.3×0.8) = 0.94. Similarly, the immediate value of performing action a_2 is (0.7×0.4)+(0.3×1.5) = 0.73. Fig. 9 (a) graphically depicts the immediate values over the belief space at the current belief state. The immediate cost (horizon length 1 value) for each action defines a linear function over belief space. We want to choose the action that gives the *lowest value* depending on the particular belief state. In the figure, we also show the partition of belief space which this value function imposes. The gray region denotes all the belief states where action a_2 is the best strategy to use while the white region is the belief states where action a_1 is the best strategy. Since the current belief state lies in the gray region, action a_2 is the best available action for belief state b.



(a) (b)

Fig. 9. A graphical representation of the belief states and value functions: (a) the current belief state and immediate values over the belief space, (b) next belief state and horizon-1 value function.

We next show how to compute the horizon 2 value of belief state b given an action a_2 and an observation o_2 (which corresponds to a high temperature reading). The horizon 2 value of a belief state is simply the value of the immediate action plus the value of the next action. In general, we would like to find the best possible value which would include considering all possible sequences of two actions. However, since in this restricted problem our immediate action is fixed, the immediate value is fully determined. The only question is what the best attainable value for the initial belief state b is when we perform action a_2 and observe o_2. We assume that with this information by using (2), the next belief state b' is computed as [0.3 0.7]. This new belief state is the belief state we are in when we have one more action to perform. We know what the best values are for every belief state when there is a single action left to perform; this is exactly what our horizon 1 value function tells us. Note that from looking at where b' is in the belief space, we immediately know that the best action we should take is a_1. Therefore, the best horizon 2 value of belief state b, given action a_2 and observation o_2, is 0.73+(0.3×1.0)+(0.7×0.8)=1.59. This value corresponds to the sequence of two actions: a_2 followed by a_1. Fig. 9 (b) illustrates the horizon-1 value function at the next belief state for initial action a_2 and observation o_2.

Next we show how to compute the value of belief state b given only an action a_2. In our problem setup, there are two possible observations o_1 and o_2. Even though we know the action with certainty, the observation we get is not known a priori. For the given belief state b, each observation has a certain probability associated with it. Since we know the value of the resulting belief state given the observation, to obtain the value of the belief state without knowing the observation, we simply weigh each resulting value by the probability that we will get that observation. Continuing with the previous example, let's assume that when we observe o_1 after action a_2, from (2), the next belief state b' is [0.6 0.4]. Looking at where b' is in the belief space, we know that the best action is a_2. Therefore, the horizon 2 value of belief state b, given a_2 and o_1, is 0.73+(0.6×0.4)+(0.4×1.5)=1.57. To summarize, starting in b and fixing the initial action to a_2, the next best action to do is a_2 if we observe o_1 and it is a_1 if we observe o_2.

Similarly, we can compute the optimal strategy for b given the initial action is a_1. More precisely, assume that if we observe o_1 after action a_1, the next belief state b' will be [0.9 0.1], whereas if we observe o_2 after action a_1, the next belief state b' is [0.5 0.5]. Then, the horizon 2 value of the belief state b when we fix the action at a_1 and observe o_1 is 0.94+(0.9×0.4)+(0.1×1.5)=1.45 corresponding to action a_2 whereas if we observe o_2 after a_1, the horizon 2 value is 0.94+(0.5×1)+(0.5×0.8)=1.84 corresponding to action a_1. To summarize, starting in b and fixing the initial action to a_1, the next best action is a_2 if we observe o_1 and it is a_1 if we observe o_2.

Suppose now the probabilities of getting observations o_1 and o_2 for the given belief state b and action a_2 are 0.45 and 0.55, respectively. These probabilities for the given belief state b and action a_1 are 0.75 and 0.25, respectively. Hence, the horizon 2 value of the belief state b when we fix the action at a_2 is (0.45×1.59)+(0.55×1.57)=1.58 and that when we fix the action at a_1 is (0.75×1.45)+(0.25×1.84)=1.55. The optimal strategy for b is the one that yields the least horizon 2 value. In this case, the strategy whereby we "do a_1 and then do a_2 if o_1 and do a_1 if o_2" is the optimal strategy for b.

Now if we fix the current action to be a_1 and the future action to be the same as it is at point b (i.e., o_1:a_2, o_2:a_1), we can find the value of every single belief point for that particular strategy. This is the best strategy to use for b, but may not be the best strategy for other points in the belief space. To efficiently compute the optimal strategy for all belief points, we utilize "transformed horizon 1 value functions" for different initial actions and partition the 1-D continuous belief space into a set of segments, where one optimal strategy holds within each segment. The value function transformation and partitioning procedure are straightforward and omitted here for brevity.

VI. Dynamic Power Management

We introduce two techniques that incorporate the proposed uncertainty management framework: *offline* and *online* DPM techniques. The offline DPM technique finds an optimal action, assuming that the inputs to the power manager are known in advance. Our approach for offline DPM is similar to conventional offline DPM techniques [19]-[21] in the sense that entire input values are known before making any decisions;

the difference is that in our offline DPM framework we consider uncertainty in reported power and delay values. On the other hand, the online DPM technique refers to strategies that attempt to find an optimal action based on information available at runtime. The proposed online DPM utilizes a Kalman filter based technique for belied state estimation to reduce the computational complexity.

A. Offline Dynamic Power Management

We construct offline a collection of policies, where a policy is a list of state-action pairs, usually implemented as a hash table with key being the state and the value being the action. Policies are generated in advance through extensive offline simulations as explained in section V. Various policies are organized into a decision tree where each leaf node represents a policy, as illustrated in Fig. 10 (a). Nodes in the decision tree are indexed by the parameters that characterize the performance state of the system, where we use the power dissipation and execution delay values, e.g., [18mW 20mw] and [4ns 8ns]. The best policy can be found by tracing the appropriate path from the root node to a leaf node in the decision tree using the given parameter values. Once a policy is located, the belief state probability is used as the key into the policy hash table to find the optimal action.

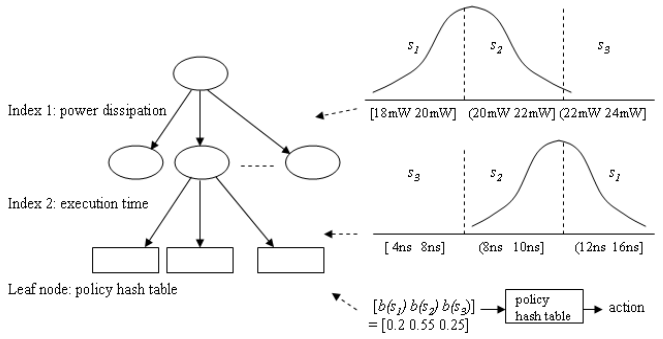


Fig. 10. (a) A decision tree of policy tables (b) probability density function of power and delay values used to trace a path from root to a leaf node in the tree.

In the aforementioned approach, we assume that power dissipation and execution times are given in the form of probability density functions (e.g., normal distribution) based on state-action pairs, as shown in Fig. 10 (b), where s_1, s_2, and s_3 are defined as, <[18mW 20mW], [4ns 8ns]>, <(20mW 22mW], (8ns 12ns]>, and <(22mW 24mW], (12ns 16ns]>, respectively. By doing so, we consider uncertainty in performance state while indexing the level of power dissipation and execution delay. For example, device power P is assumed to be a normally distributed random variable with a mean value of P_{sim} and a standard deviation of ΔP induced by uncertainty, as illustrated in top of Fig. 10 (b).

In our problem setup, P_{sim} is the simulated power number while ΔP is the standard deviation of power values, which is calculated by running different tasks on the system at different process corners (e.g., fast, typical, and slow) available with the TSMC 65nm library. Furthermore, we can vary the ranges of power values for states (e.g., range of 2mW in [18mW 20mW] can be changed to the range of 4mW resulting in [17mW 21mW]), considering a higher standard deviation (i.e., uncertainty). The execution times are treated in the same way. Then, belief state which represents the probabilities of being in each of the performance states is obtained as a key to policy hash table. For example, referring to Fig. 10 (b), suppose that the probabilities of being in s_1, s_2, and s_3 are 0.3, 0.6, and 0.1 in terms of the power dissipation level, and 0.1, 0.5, and 0.4 in terms of the execution delay. Then, the belief state [$b(s_1)$ $b(s_2)$ $b(s_3)$] is calculated simply as [0.2 0.55 0.25] by taking the average value of the two probability vectors.

Fig. 11 summarizes the offline power management technique with a decision tree-based policy selection, where power and delay values are given as $N(P_{sim}, (\Delta P)^2)$ and $N(D_{sim}, ((\Delta D)^2)$. Similar to the power values, D_{sim} denotes the simulated delay number while ΔD is the standard deviation of delay values. When the power manager receives a performance state with the knowledge of previously assigned action-state pairs, an optimal action is selected by the PM based on the policy hash table, and issued to the system, which causes the system state to change.

input: $N\left(P_{sim},(\Delta P)^2\right)$, $N\left(D_{sim},(\Delta D)^2\right)$
output: *action*
1: index level of power dissipation
2: index level of execution delay
3: achieve belief state based on indexes
4: find an appropriate action through hash table
5: return *action*

Fig. 11. An offline power management technique.

B. Online Dynamic Power Management

For an online power management, belief-state transition probabilities are not given in advance. Note that the complexity of computation required by Eqn. (2) for updating the belief state grows rapidly with the number of state variables, making it infeasible for real-time applications, e.g., online DPM techniques. In addition, calculating exact solutions for the finite-horizon stochastic POSMDP problems is P-SPACE hard [11]. Therefore, exact solutions cannot be found for belief-state SMDP with more than a handful of states. Indeed, solving a belief-state SMDP problem is extremely expensive because of the complexity of calculating the exact belief state [38]. To overcome this difficulty, one is usually forced to estimate the system state by some other approaches. By doing so, the overwhelming complexity in deriving a power management policy for every possible situation is avoided.

The basic idea of our online power management technique is to use the estimation of the unknown state based on a look-ahead search technique which also includes a step to predict an unknown error while estimating. Hence, we interleave state estimation based on "*Kalman filter*" technique [39] and policy optimization based on the value iteration algorithm. Details are provided below.

We present a prediction-based online DPM technique, which is analytically and statistically tractable. First, assuming

that we know the distribution of PVT variation and observation noise, we can define the state and observation models simply in accordance with our proposed framework as follows:

$$b^{t+1} = \mathbf{X}b^t + \mathbf{Y}a^t + u^t, \quad u^t \sim N(0, Q^t) \tag{8}$$

$$o^{t+1} = \mathbf{Z}b^{t+1} + v^{t+1}, \quad v^{t+1} \sim N(0, R^t) \tag{9}$$

where t denotes a time step, u^t is a state noise induced by PVT variation which is normally distributed with zero mean and variance Q^t, v^{t+1} is a temperature observation noise normally distributed with zero mean and variance R^t. The state transition matrix **X** includes the probabilities of transitioning from state b^t to another state b^{t+1} when action at is taken, the action-input matrix **Y** relates the action input to the state, whereas the observation matrix **Z**, which maps the true state space into the observed space, contains the probabilities of making observation o^{t+1} when action a^t is taken, leading the system to enter state s^{t+1}. In practice, **X**, **Y**, and **Z** might change with each time step or measurement, but here we assume they are constant.

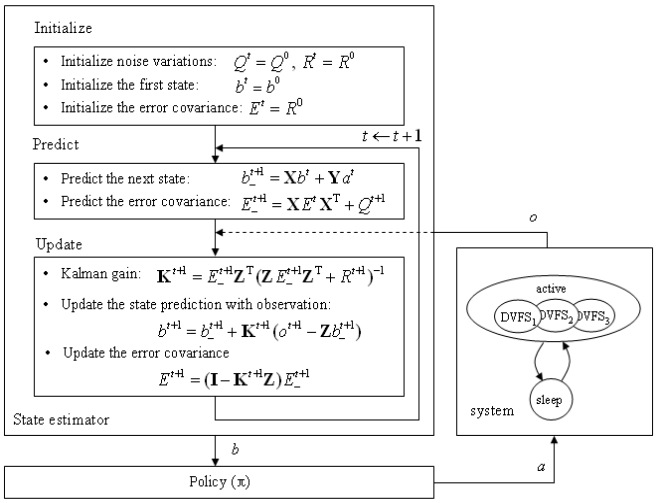


Fig. 12. The structure of online power management.

With above-mentioned parameters, the structure of our proposed online DPM is provided in Fig. 12[3]. The estimation algorithm performs the state estimation based on KF as follows.

a) Initialize: The algorithm initializes the first state b^t as b^0, and the error covariance matrix E^t, which is a measure of the estimated accuracy of the state prediction, to a diagonal matrix where the diagonal elements are set to some fixed value, signifying that the initial system state is uncertain.

b) Predict: The algorithm computes the predicted (a priori) state b_-^{t+1} and the predicted (a priori) error covariance matrix E_-^{t+1}.

c) Update: The algorithm first computes the optimal Kalman gain $\mathbf{K}^{t+1}$ and uses it to produce an updated (a posteriori) state estimate, b^{t+1}, as a linear combination of b_-^{t+1} and the Kalman gain-weighted residue between an actual observation o^{t+1} and the predicted observation $\mathbf{Z}b_-^{t+1}$. The algorithm also updates the error covariance matrix.

This iterative approach is one of the appealing features of the Kalman filter.

Simply speaking, the proposed online DPM technique estimates the next belief state based on the KF technique, and computes the belief-state transition probabilities and observation functions by simply deriving the maximum likelihood estimates, while storing the occurrence frequencies. Fig. 13 shows the proposed online DPM technique based on the Kalman filter technique, where an appropriate action is given to the system by utilizing the value iteration algorithm (see Fig. 7) after estimating the belief state.

input: *o*
output: *action*
1: observe temperature *o*
2: estimate the next state *b'*
3: compute T_b and *Z* by deriving maximum likelihood estimates
4: find an appropriate action through value iteration algorithm
5: return *action*

Fig. 13. An online power management technique.

VII. Experimental Results

In the experimental setup, we implemented a 32bit RISC processor compatible with [40] in TSMC 65nmLP library, which has 3 optional operating voltages (1.08V, 1.20V, and 1.29V) and dual threshold voltages. We developed the proposed framework in Matlab, which allows us to rapidly consider multiple scenarios with respect to the magnitude and distribution of PVT variations.

To achieve accurate power values for dynamic power and leakage power consumption, we first generated a forward SAIF (Switching Activity Interchange File) after synthesizing into gate-level netlist. Second, we obtained a backward SAIF by back-annotated RTL simulation with the Specman function simulator [41], and then executed the Power Compiler [29], where the switching activities of the netlist are incorporated so as to calculate accurately the dynamic and static power consumption (cf. Fig. 14).

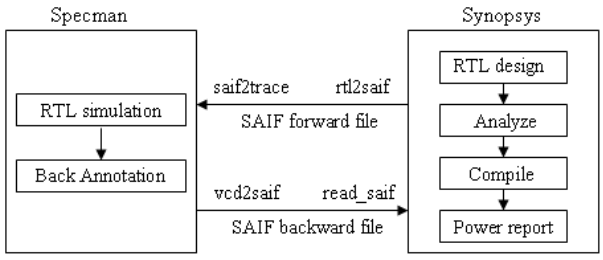


Fig. 14. Flow of power simulation.

In the first experiment, we analyzed characteristics of the designed processor in terms of power dissipation by executing

[3] The subscript "-"denotes that the value calculated at the prediction stage will be updated in the correction stage.

SPECint2000 benchmark programs [42] where we include data for only three of the benchmark programs: *gcc*, *gap*, and *gzip*. TABLE II reports the power dissipation distribution of the processor, indicating that certain components of the processor such as the execution units and the register units have a very high power density. Fig. 15 shows leakage power variation on the processor, obtained by varying the process corner cases.

TABLE II
THE DISTRIBUTION IN PERCENTAGE OF POWER DISSIPATION IN THE PROCESSOR (NO CACHE)

SPECint 2000	Function blocks												
	dOutreg	dAreg	iAreg	incr	mul	shifter	alu	sr	reg	decode	fetch	execute	busCtl
gcc	4.6	14.4	13.8	4.1	2.3	2.7	16.5	2.2	15.4	4.1	4.1	4.1	11.7
gap	4.3	13.1	15.7	4.0	4.2	3.1	14.2	1.7	14.8	4.2	4.2	4.2	12.3
gzip	4.6	9.2	15.4	4.6	4.5	3.8	18.6	2.3	15.1	4.6	4.6	4.6	8.1

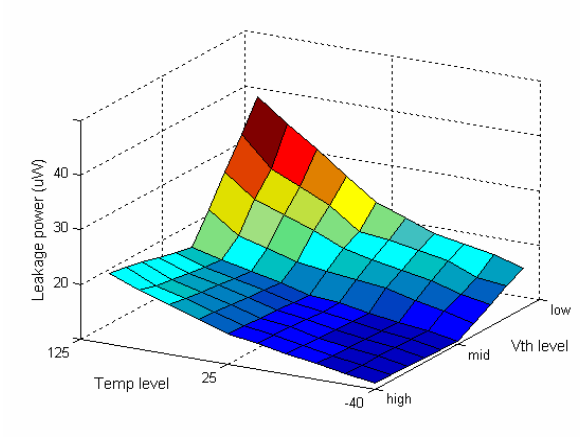


Fig. 15. Leakage power for different levels of variability.

The second experiment is to demonstrate the effectiveness of the proposed DPM under uncertainty management framework. First, we set the parameter values for the evaluation of the proposed framework as shown in TABLE III, where we have sets of three actions $\{a_1, a_2, a_3\}$, where a_1 = [1.08V / 500MHz], a_2 = [1.20V / 650MHz], and a_3 = [1.29V / 800MHz], and observations $\{o_1, o_2, o_3\}$. The range of observations is defined by the temperature thresholds based on the ACPI (Advanced Configuration and Power Interface) specification [43]. The expected cost rate is defined as the power-delay product (PDP) of the processor for each state and action pair, where we set the range of performance states $\{s_1, s_2, s_3\}$ as a combination of power dissipation and execution delay values for the processor. For example, cost $k(s_1, a_1)$ is the power-delay product of the system that stays in state s_1 when action a_1 is taken, i.e., 18mW (least power) × 12nS (highest delay) = 216pJ. Similarly, $k(s_1, a_2)$ and $k(s_1, a_3)$ are calculated as 20.75mW (medium power) × 10.5nS (medium delay) ≈ 218pJ, and 23.5mW (highest power) × 9nS (least delay) = 212pJ, respectively. Note that we define different cost values for a system state (e.g., s_1), since different actions (e.g., a_1, a_2, or a_3) can cause the system to transition into the same system state (i.e., the system maintains the same range of performance values) with difference cost values. We achieved these values by running the Power Compiler while varying the levels of operating voltage and frequency.

TABLE III
PARAMETER VALUES FOR A GIVEN EXPERIMENT

State	Description				Observation	Description [°C]		cost $k(s, a)$			
	delay (ns)		power (mW)						a_1	a_2	a_3
s_1	(9.0	12.0]	[18.0	23.5]	o_1	[50	65]	s_1	216	218	212
s_2	(6.5	9.0]	(23.5	25.5]	o_2	(65	75]	s_2	212	190	165
s_3	[3.5	6.5]	(25.5	30.5]	o_3	(75	90]	s_3	165	138	105

We arbitrarily chose a sequence of 50 application program runs, comprising of instances of *gcc*, *gap*, and *gzip* benchmarks, e.g., gap_1 - $gzip_2$ - gap_3 - gcc_4 -…- gap_{50}, where $program_i$ is the i-th program in the sequence. The sequence of 50 application programs is executed on the processor to calculate the belief states based on the estimated temperature which serves as the observation. Because we do not have a packaged IC equipped with a thermal sensor to report the on-chip temperature, we estimate the on-chip temperature by utilizing

$$T_{chip} = T_A + P \cdot (\theta_{JA} - \psi_{JT}) \quad (10)$$

based on the parameter values extracted from the commercial data sheet for a PBGA package [44]. Note that T_A is the ambient temperature, θ_{JA} is the thermal resistance for junction-to-ambient, ψ_{JT} denotes the junction-to-top of package thermal characterization parameter, and P is the power dissipation. Next, the belief states are evaluated based on the actions and observations over the state space as the processor executes the sequence of programs.

Fig. 16 shows the trace of belief state for state s_1, s_2, and s_3, where we use the Kalman filter estimation technique of the proposed online DPM framework. We set that the values of PVT variation variance Q and observation noise variance R equal to 1.1, where we achieved the probability density function for the power consumption of the processor such that the mean value is 25mW and covariance is 1.1 (i.e., N(25 1.1). In our experiment, the time steps are abstractly defined and the power manager issues a command at each time step (i.e., decision epochs), where observations are made when each program in the sequence has been completed.

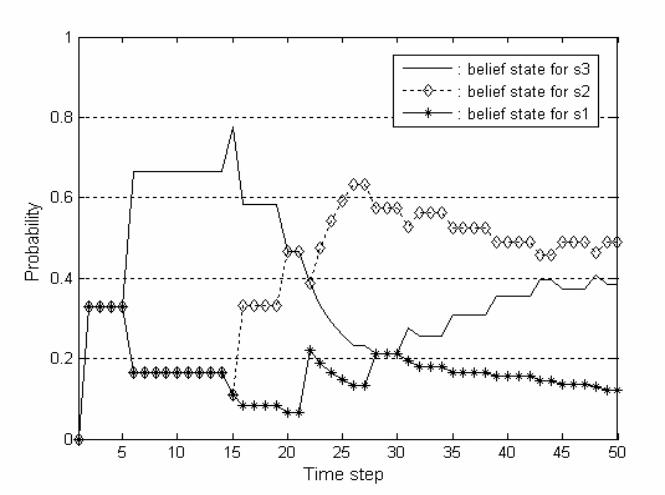


Fig. 16. Trace of belief state for state estimation.

Simulations reported in Fig. 17 show the results of the policy generation algorithm based on the information provided in TABLE III and Fig. 16. We set the discount factor as 0.5 when evaluating the value function. The optimal action is chosen to minimize the value function.

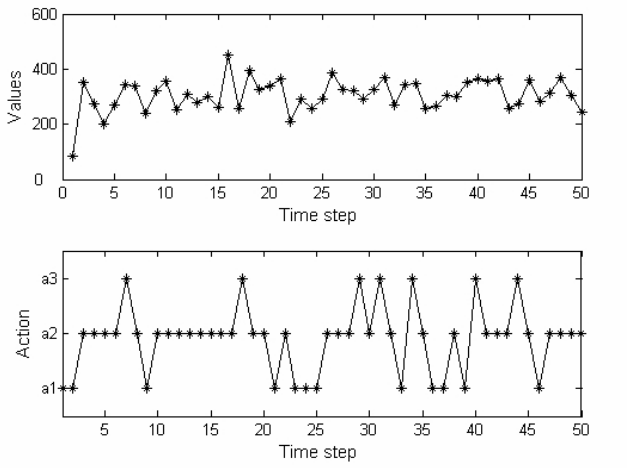


Fig. 17. Evaluation of policy generation algorithm.

In the third experiment, we investigate how robustly the proposed approach can handle variability during the power management process by comparing with various operating conditions (i.e., worst and best corners). The optimal DPM policy is achieved by evaluating the value function with the derived state transition probabilities. In our approach, we performed tasks while varying the operating conditions, and identifies the most probable system state given noisy temperature observations. Table IV summarizes these simulation results in terms of power, energy, and normalized energy-delay-product (EDP) as the figure of merit. Clearly, the uncertainty-aware DPM approach cannot do any better than a conventional DPM at the best corner case. The expectation, however, is that it will outperform the conventional DPM at the worst corner case, while ensuring energy efficiency. It is also clearly seen that a lot of silicon performance is left untapped under the worst corner-case assumption.

Table IV
COMPARING RESULTS OF OUR PROPOSED APPROACH WITH THE CORNER-BASED RESULTS

Condition	Average power (mW)	Average energy (mJ)	EDP (normalized)
Our approach	25.8	163.5	1.27
Worst corner	22.7	197.2	2.04
Best corner	28.8	155.4	1.00

The fourth experiment is designed to evaluate the proposed offline/online DPM techniques by capturing the energy-saving opportunities of the system. In both DPM policies, we compare the performance of the proposed technique with a conventional DPM approach, similar to that presented in [45], which can be defined simply as follows (denoted by Greedy), where $DVFS_1 < DVFS_2 < DVFS_3$ in terms of operating voltage and frequency values.

Greedy: Apply the following DPM strategy.

- When the workload of tasks (e.g., the arrival rate of tasks) is low, we use the lowest $DVFS_1$ value.
- When the workload of tasks is high, we use the highest $DVFS_3$ value. Otherwise, we use the $DVFS_2$ value.

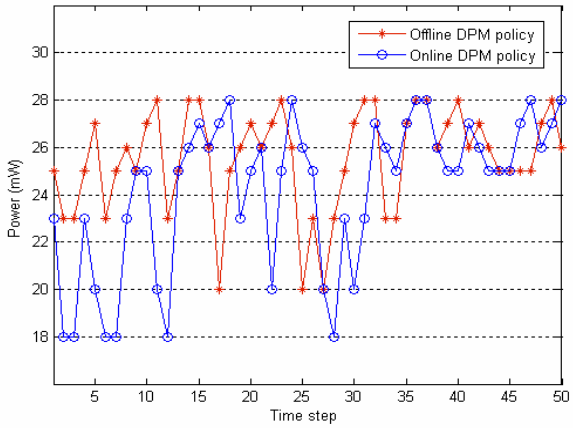


Fig. 18. Power consumption of offline / online DPM policies.

Note that we consider the overhead of power-mode transitions during simulation as illustrated in [46]. The simulation results for the proposed offline and online DPM technique are shown in Fig. 18, where the online DPM policy tends to dynamically adapt to environmental changes, which can incur a mode transition penalty. This is because the online DPM algorithm performs prediction-correction procedure to react to the environmental changes. Table V presents the simulation results in terms of power savings (%). It includes the specific power saving result for each performance state. For example, there is 10.5% power savings in the state s_3 by the offline DPM, whereas there is 7.4% power penalty in the state s_1. The table shows that the proposed DPM policies result in power savings when the system is in the state s_2 and s_3. However, there is no significant impact for online DPM in terms of average power savings.

Table V
COMPARISON OF OUR DPM POLICIES WITH THE CONVENTIONAL APPROACH IN TERMS OF POWER SAVINGS

Greedy	Proposed technique				
avg. power (mW)	DPM policy	power saving in each state (%)			average power saving (%)
		s_1	s_2	s_3	
24	Offline DPM policy	-7.4	3.1	10.5	2.1
	Online DPM policy	-9.2	1.7	8.9	0.5

Table VI gives the result of the energy savings by the proposed DPM techniques. Contrary to the little impacts on power savings, this result demonstrates that our approaches greatly reduce the total energy dissipation especially in the state s_1 and s_2. For example, there is 21.1% energy savings in state s_1 by the offline DPM policy, although we have 7.4% power penalty by running the same DPM policy. The conventional DPM approach (which is unaware of the PVT variations), however, can outperform slightly our DPM technique in terms of energy savings only for the case that the system is in state s_3, where our online DPM technique produces an *a priori* estimate for the next time step which may result in energy waste.

Overall, the proposed DPM techniques achieve energy savings in the presence of the PVT variations up to average of 10.3% and 6.8% in the case of offline and online policies, respectively. Furthermore, it is clearly seen that if we focus on conserving energy in low performance settings, we can achieve energy saving up to 21.1%, and 16.8% in the case of offline and online policies, respectively (see energy saving in state s_1). This scenario typically occurs in applications that require low voltage-frequency value for their operations.

Table VI
COMPARISON OF OUR DPM POLICIES WITH THE CONVENTIONAL APPROACH IN TERMS OF ENERGY SAVINGS

Greedy	Proposed technique				
avg. energy (mJ)	DPM policy	energy saving in each state (%)			average energy saving (%)
		s_1	s_2	s_3	
175	Offline DPM policy	21.1	12.3	-2.5	10.3
	Online DPM policy	16.8	8.1	-4.5	6.8

VIII. CONCLUSION

We addressed the problem of system-level power management subject to variability in system performance parameters and uncertainty in observations made on the system. In particular, we presented a system-level power management approach based on a stochastic decision making framework i.e., a partially observable Markovian decision process model, which is capable of coping with uncertainty in system state and observations. This uncertainty management framework guarantees to find an optimal power management policy by utilizing a value iteration algorithm. We implemented both offline and online DPM techniques and reported experimental results demonstrating their effectiveness in robustly reducing total system energy dissipation when running a variety of applications.

REFERENCES

[1] K. Bowman, S. Duvall, and J. Meindl, "Impact of die-to-die and within die parameter fluctuations on the maximum clock frequency distribution," Proc. of International Solid-State Circuits Conference, 2001, pp. 278–279.

[2] S. Mukhopadhyay and K. Roy, "Modeling and estimation of total leakage current in nano-scaled CMOS devices considering the effect of parameter variation," Proc. of Symposium on Low Power Electronics and Design, Aug. 2003, pp. 172–175.

[3] S. Abbaspour, H. Fatemi, and M. Pedram, "Parameterized block-based non-Gaussian variational gate timing analysis," IEEE Trans. on Computer Aided Design, Vol. 26, No. 8, Aug. 2007, pp. 1495-1508.

[4] J. Tschanz, J. Kao, S. Narendra, R. Nair, D. Antoniadis, A. Chandrakasan, and V. De, "Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage," Journal of Solid State Circuits, vol. 37, no. 11, 2002, pp. 1396–1402.

[5] Y. Komatsu, K. Ishibashi, et al., "Substrate-noise and random-fluctuations reduction with self-adjusted forward body bias," Proc. of Custom Integrated Circuits Conference, 2005, pp. 35–38.

[6] M. Mani, A. Singh, and M. Orshansky, "Joint Design-Time and Post-Silicon Minimization of Parametric Yield Loss using Adjustable Robust Optimization," Proc. of International Conference on Computer Aided Design, 2006, pp. 19-26.

[7] N. Azizi, M. M. Khellah, V. De, and F. N. Najm, "Variations-aware low-power design with voltage scaling," Proc. of Design Automation Conference, 2005, pp. 529–534.

[8] D. Marculescu and E. Talpes, "Variability and energy awareness: a microarchitecture-level perspective," Proc. of Design Automation Conference, 2005, pp. 11–16.

[9] N. S. Kim, T. Kgil, K. Bowman, V. De, and T. Mudge, "Total power optimal pipelining and parallel processing under process variations in nanometer technology," Proc. of Int'l Conference on Computer Aided Design 2005, pp. 535–540.

[10] A. Srivastava, D. Sylvester, and D. Blaauw, Statistical Analysis and Optimization for VLSI: Timing and Power. Springer, 2005.

[11] M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Publisher, New York, 1994.

[12] H. Jung, and M. Pedram, "Dynamic Power Management under Uncertain Information," *Proc. of Design Automation and Test in Europe*, Apr. 2007, pp.1060 - 1065.

[13] Y.F. Tsai, N. Vijaykrishnan, Y. Xie, and M.J Irwin, "Influence of Leakage Reduction Techniques on Delay/Leakage Uncertainty," Proc. of IEEE 18th Int'l Conference on VLSI Design, Jan. 2005, pp.374-379.

[14] H. Su, F. Liu, A. Devgan, E. Acar, and S. Nassif, "Full Chip Leakage Estimation Considering Power Supply and Temperature Variations," Proc. of International Symposium on Low Power Electronics and Design, Aug. 2003, pp. 78-83.

[15] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Kehavarzi, and V. De, "Parameter Variations and Impact on Circuits and Microarchitecture," Proc. of Design Automation Conference, Jun. 2003, pp.338-342.

[16] M. Lie, W.S. Wang, and M. Orshansky, "Leakage Power Reduction by Dual-Vth Designs Under Probabilistic Analysis of Vth Variation," Proc. of International Symposium on Low Power Electronics and Design, Aug. 2004, pp. 2-7.

[17] A. Basu, S-C. Lin, V. Wason, A. Mehrotrat, and K. Nanerjee, "Simultaneous Optimization of Supply and Threshold Voltages for Low-Power and High-Performance Circuits in the Leakage Dominant Era," Proc. of Design Automation Conference, Jun. 2004, pp. 884-887.

[18] L. Benini, and G. De Micheli, *Dynamic Power Management: Design Techniques and CAD Tools*, Kluwer Academic Publishers, 1998.

[19] L. Benini, G. Paleologo, A. Bogliolo, and G. De Micheli, "Policy Optimization for Dynamic Power Management," IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems, Vol. 18, Issue 6, pp. 813-833, Jun. 1999.

[20] Q. Qiu, Q. Wu, and M. Pedram, "Stochastic Modeling of a Power-Managed System – Construction and Optimization," IEEE Trans. on Computer-Aided Design, Vol. 10, No. 10, pp. 1200-1217, Oct. 2001.

[21] T. Simunic, L. Benini, P. Glynn, and G. De Micheli, "Event-driven Power Management," IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems, Vol. 20, Issue 21, pp. 840-857, Jul. 2001.

[22] Z. Ren, B. H. Krogh, and R. Marculescu, "Hierarchical Adaptive Dynamic Power Management," IEEE Trans. on Computer, Vol. 15, Issue 4, pp. 409-420, Apr. 2005.
[23] P. Rong and M. Pedram, "Hierarchical dynamic power management with application scheduling," Proc. of Symposium on Low Power Electronics and Design, Aug. 2005, pp. 269-274.
[24] Y.F. Tsai, N. Vijaykrishnan, Y. Xie, and M.J Irwin, "Influence of Leakage Reduction Techniques on Delay/Leakage Uncertainty," *Proc. of IEEE 18th International Conference on VLSI Design*, Jan. 2005, pp.374-379.
[25] H. Su, F. Liu, A. Devgan, E. Acar, and S. Nassif, "Full Chip Leakage Estimation Considering Power Supply and Temperature Variations," *Proc. of International Symposium on Low Power Electronics and Design*, Aug. 2003, pp. 78-83.
[26] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Kehavarzi, and V. De, "Parameter Variations and Impact on Circuits and Microarchitecture," *Proc. of Design Automation Conference*, Jun. 2003, pp.338-342.
[27] A. Srivastava, D. Sylvester, and D. Blaauw, *Statistical Analysis and Optimization for VLSI: Timing and Power*, Springer, 2005.
[28] Synopsys Prime Time. http://www.synopsys.com.
[29] Synopsys Power Compiler Documents. http://www.synopsys.com.
[30] M. Pedram, and S. Nazarian, "Thermal Modeling, Analysis, and Management in VLSI Circuits: Principles and Methods," *Proc. of IEEE, Special Issue on Thermal Analysis of ULSI*, Vol. 94, No. 8, pp. 1487-1501, Aug. 2006.
[31] K. Skadron, M.R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-Aware Microarchitecture," *Proc. of International Symposium on Computer Architecture*, Jun. 2003, pp. 2-13.
[32] A.R. Cassandra, L.P. Kaelbling, and M.L. Littman., "Acting Optimally in Partially Observable Stochastic Domains," *Proc. of 12th National Conference on Artificial Intelligence*, Aug. 1996, pp. 1023-1028.
[33] R.D. Smallwood and E.J. Sondik, "The optimal control of partially observable Markov decision processes over a finite horizon," *Operations Research*, 21, 1071-1088, 1973.
[34] K. J. Astron, "Optimal Control of Markov Decision Processes with Incomplete State Estimation," *Journal of Mathematical Analysis and Application*, Vol. 10, pp. 174-205, 1965.
[35] A Gosavi, *Simulation-based Optimization: Parameter Optimization Techniques and Reinforcement Learning*, Kluwer Publishers, 2003.
[36] R.E. Bellman, *Dynamic Programming*. Princeton University Press, Princeton, 1957.
[37] R. Williams, and L. Baird, "Tight Performance Bounds on Greedy Policies based on Imperfect Value Functions," *Technical report* NU-CCS-93-14, Northeastern University, Nov. 1993.
[38] S. Paquet, G. Gordon, and S. Thrun, "Point-based Value Iteration: An Anytime Algorithm for POMDPs," *Proc. of International Joint Conference on Artificial Intelligence*, Aug. 2003, pp.1025-1032.
[39] R.E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Trans. of the ASME – Journal of Basic Engineering*, Vol. 82, Series D, pp. 35-45, 1960.
[40] OpenRISC 1000 processor. http://www.opencores.org.
[41] Specman HDL simulator. http://www.cadence.com.
[42] CPU SPECint2000 document. http://www.spec.org.
[43] Advanced Configuration and Power Interface Specification, Rev. 3.0b, Oct. 2006. http://www.acpi.info/spec.htm.
[44] Y. Cheng, C. Tsai, C. Teng, and S. Kang, *Electrothermal Analysis of VLSI Systems*. Kluwer Academic Publishers, 2000.
[45] R. Jejurikar, and R. Gupta, "Dynamic Voltage Scaling for System-wide Energy Minimization in Real-time Embedded Systems," *Proc. of International Symposium on Low Power Electronics and Design*, Aug. 2004, pp. 78 - 81.
[46] T. D. Burd, and R. W. Brodersen, "Design Issues for Dynamic Voltage Scaling," Proc. of *International Symposium on Low Power Electronics and Design,* Aug. 2000, pp. 9 - 14.

Hwisung Jung (S' 99) received B.S. and M.S. degree in electrical engineering from Yonsei University, Korea in 1996 and 2001, respectively. He is currently pursuing the Ph.D. degree in electrical engineering at the University of Southern California, Los Angeles. He was with LG semiconductor and Samsung electronics from 1995 to 1999 and from 2001 to 2003, respectively, where he worked as a research engineer in the field of low-power ASIC design. He was also with Broadcom as a research intern during the summer of 2006 and 2007. He has authored more than 10 papers in the several conferences, including Design Automation Conference, Design and Test in Europe, and Asia-Pacific Design Automation Conference.

His research interests are in the area of system-level power management and low-power design.

Massoud Pedram (S'88 – M'90 – SM'98 – F'01) received B.S. degree in Electrical Engineering from the California Institute of Technology in 1986. Subsequently, he received M.S. and Ph.D. in Electrical Engineering and Computer Sciences from the University of California, Berkeley in 1989 and 1991, respectively. In September 1991, he joined the Department of Electrical Engineering at the University of Southern California where he is currently a professor and Chair of the Computer Engineering division. Dr. Pedram is a recipient of the National Science Foundation's Young Investigator Award (1994) and the Presidential Early Career Award for Scientists and Engineers (a.k.a. the Presidential Faculty Fellows Award) (1996). His research has received a number of awards including two Best Paper Awards from the International Conference on Computer Design, two Design Automation Conference Best Paper Awards, an IEEE Transactions on VLSI Systems Best Paper Award, and an IEEE Circuits and Systems Society Guillemin-Cauer Award.

Dr. Pedram has served on the technical program committee of a number of conferences and workshops, including Design Automation Conference (DAC), Design and Test in Europe (DATE), Asia-Pacific Design Automation Conference (ASP-DAC), International Conference on Computer Aided Design (ICCAD), International Symposium on Low Power Electronics and Design (ISLPED), International Symposium on Physical Design (ISPD), and International Workshop on Logic Synthesis (IWLS). Dr. Pedram was a co-founder and general chair of the 1995 International Symposium on Low Power Design and the technical co-chair and general co-chair of the 1996 and 1997 International Symposium on Low Power Electronics and Design, respectively. He was the Technical Chair of the 2002 International Symposium on Physical Design and is the General Chair of the 2003 symposium. Dr. Pedram has given several tutorials on low power design at major CAD conferences and forums including, DAC, ICCAD, and ASP-DAC. He has published more than 300 journal and conference papers and written four books on various aspects of low power design.

Dr. Pedram is an IEEE Fellow and an ACM member and currently serves on the Advisory Board of the ACM Special Interest Group on Design Automation. He served as an Associate Editor of the IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems and ACM Transactions on Design Automation of Electronic Systems. He received the 2000 Distinguished Service Award of ACM - SIGDA for contributions in developing the SIGDA Multimedia Monograph Series and organizing the Young Student Support Program. Dr. Pedram was a member of the Board of Governors of the IEEE Circuits and Systems Society from 2000 to 2002, Chair of the Distinguished Lecturer Program of the IEEE CASS for 2003 and 2004, and the CASS VP of Publications in 2005 and 2006.