Dynamic Power Management under Uncertain Information

Hwisung Jung, Massoud Pedram
Department of Electrical Engineering, University of Southern California
Los Angeles, CA 90089
{hwijung, pedram}@usc.edu

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, voltage, and temperature (PVT) variations and fluctuations. This uncertainty significantly undermines the accuracy and effectiveness of traditional DPM approaches. More specifically, we propose a stochastic framework to improve the accuracy of decision making in power management, while considering the manufacturing process and/or design induced uncertainties. A key characteristic of the framework is that uncertainties are effectively captured by a partially observable semi-Markov decision process. As a result, the proposed framework brings the underlying probabilistic PVT effects to the forefront of power management policy determination. Experimental results with a RISC processor demonstrate the effectiveness of the technique and show that our proposed variability-aware power management technique ensures robust system-wide energy savings under probabilistic variations.

1. Introduction

In the nanometer era, PVT variations, especially within-chip variations, pose a major challenge to the design of low-power circuits and systems. These within-chip variations that arise either from environmental variations (i.e., temperature or voltage) or manufacturing process variations (i.e., dopant fluctuations, T_{ox} and L_{eff} variability) can result in uncertainty in the power and delay estimation [1][5]. Variability refers to the known quantitative relationship to a source, whereas uncertainty refers simply to something we cannot describe deterministically precisely. Modern VLSI circuits are becoming increasingly sensitive to the rising levels of variability in process and design parameters. Lack of proper modeling and analysis tools transforms variability to uncertainty [6]. Thus, integrating different sources of uncertainty in one conceptual framework is becoming challenging.

As evidenced in the recent literature [1]-[5], increasing interest has focused on reducing the variability and/or uncertainty in the designs. The work presented in [1] studies the impact of leakage reduction techniques on the delay uncertainty. Since leakage is critically dependent on operating temperature and power supply, reference [2] presents a full chip leakage estimation technique which accurately accounts for power supply and temperature variations. In reference [3], the authors discuss process, voltage and temperature variations and their impact on circuit and microarchitectures beyond the 90nm technology node. Probabilistic models used to account for the impact of threshold voltage variations on the leakage power are introduced in [4]. These models are then used to minimize leakage power, while satisfying certain performance requirements. Reference [5] 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.

Most of the previous work on variability and uncertainty has focused on the variability modeling and analysis, and variation control at the circuit level and with respect to optimization problems in the physical design domain. It is important to account for different sources of variability or uncertainty early in the design process, especially at the level of developing resource management and power control strategies for large complex electronic systems. Variations (even systematic ones) at that level often translate to uncertainty because the underlying detailed circuit-level realization is not available and hence, the variations appear as imperfect or noisy (hence, uncertain) observations about the state of a system. The influence of uncertainty about the measured parameters of the system must be modeled stochastically and utilized to determine uncertainty in the performance parameters of interest. To the best of our knowledge, no proposed research work has been conducted on power management techniques with stochastic modeling for the uncertainty. Improving the accuracy and robustness of decision making by modeling and assessing the uncertainty is an important step to guarantee the quality of all kinds of system-level resource management, including dynamic power management.

In this paper, we propose a stochastic uncertainty management framework and integrate it with system-level power control. Our proposed framework is based on i) partially observable Markov decision process [7] to model the uncertainty and ii) semi-Markov decision process to model the decision making for optimizing the power consumption under a delay constraint. Markov decision process model offers a robust theoretical framework which enables one to apply strong mathematical optimization techniques in order to derive optimal policies. Finally, we present a variability-aware dynamic power management technique to illustrate the effectiveness of the uncertainty management framework.

The remainder of this paper is organized as follows. Section 2 provides a background of the paper. The details of the stochastic uncertainty management framework are given in section 3. Section 4 presents a variability-aware dynamic power management technique. Experimental results and conclusion are given in section 5 and section 6.

2. Background

As the leakage power dissipation is becoming an important portion of total power of a system beyond 130nm technology, *process*, *voltage*, and *temperature* (PVT) variations within-chip have to be controlled to reduce the impact of parameter variations. Since both temperature and supply voltage has very strong locality within-die, the variation of these results in uneven distribution of power dissipation and rapid rise of power density, which occur temperature hot spots. Note that there is a feedback loop between leakage power and temperature, since subthreshold leakage power is exponentially dependent on temperature [10]. Furthermore, temperature variations have significant impact on delay. Thus, performance estimation is not accurate without considering temperature dependency of transistor's carrier mobility. Considering voltage variations, there is some voltage IR drop on the power supply network, which result in 5~10% variations in worst-case [8].

As a result, both power dissipation and performance, affected by PVT variations, cannot be identified easily at runtime by a system itself. Therefore, we use an observation strategy (e.g., thermal-observed) in uncertainty environment to identify the profile of the system in a stochastic manner. Note that even if power density determines temperature, a change in temperature does not correspond to instantaneous power dissipation because of a low-pass filtering in translating power variations into temperature variations [11]. However, it will not hurt the quality of the paper if we assume that instantaneous power density can serve as a proxy for temperature variations, provided that multiple on-chip temperature sensors can provide information about the spatial temperature gradients in different zones of the chip [12].

The growing extent of the uncertainty in optimization problem can be attributed to the fact that the nanometer technology is approaching the regime of fundamental randomness in the behavior of silicon structures [6]. A move to stochastic optimization would mean that we need to treat many design optimization problems as random values to be described by probabilistic distributions. In some sense, treating the uncertainty probabilistically means that even after manufacturing the chip there is still uncertainty about the performance behavior, for example. Thus, an approach to solve this problem is to provide a unified stochastic management framework of the various sources of uncertainty in predicting the performance behavior of the system. Furthermore, such a stochastic framework must provide the computational tractability of the randomness to treat the many sources of uncertainty in optimizing the performance.

3. A Stochastic Decision Making Framework for Managing Uncertainty

We present a theoretical framework to construct a power management process under uncertain information.

3.1 Partially Observable Environments

The uncertainty in parameter observation, where a power manager cannot reliably identify the performance (e.g., delay and power) state of the chip (due to non-uniform distribution of variations) during power management, may be addressed by modeling decision making by stochastic processes. Note that a power manager, which observes an on-chip temperature and issues commands (or actions) to control uncertain performance state of the system, makes a decision at each event occurrence (e.g., time-based or interrupt-based), called a *decision epoch*. These actions and performance states determine the probability distribution over possible next states. Thus, the sequence of performance states of the system can be modeled as a stochastic process.

In partially observable environments, observations made by an agent (e.g., the power manager) about the state of the environment (e.g., performance) may be noisy and provide incomplete information. The most naive strategy for dealing with partial observability is to ignore it. That is, to treat the observations as if they were the actual states of the environment and act on them. This approach will result in unexpected performance behaviors by making decisions based on an erroneous Markovian process model. Some improvement is achieved by considering stochastic policies, i.e., mappings from observations to probability distributions over actions. The only way to behave truly effectively in a wide-range of environments is to use memory of previous actions and observations to disambiguate the current state. A better strategy consists of using hidden Markov model techniques to learn a model of the environment, including the hidden state, then to use that model to construct a *perfect memory* controller.

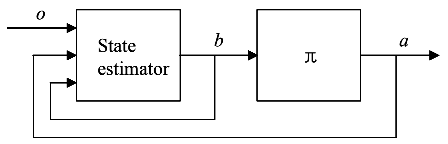

Figure 1. Structure of a POSMDP agent.

Figure 1 illustrates the basic structure for a perfect-memory controller. The component on the left is the *state estimator*, which computes the agent's *belief state*, b as a function of the old belief state, the last action a derived from policy π, and the current observation o. In this context, a belief state is a probability distribution over states of the environment, indicating the likelihood that, given the agent's past experience, the environment is actually in each of those states.

We use a semi-Markov decision process (SMDP) to model the event-driven decision making progression, but also combine it with a partially observable Markov decision process (POMDP) to consider the uncertainty in parameter observation. Notice that the time spent in a particular state in the SMDP (i.e., the sojourn time or the time difference between successive decision epochs) follows an arbitrary probability distribution, which is a more realistic assumption than an exponential distribution [7]. A partially observable semi-Markov decision process (POSMDP) extends the SMDP model by incorporating an observation model, which is defined as follows.

Definition 1: Partially Observable Semi-Markov Decision Process. A POSMDP is a tuple (S, A, O, T, C, Z) 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. $T: S \times A \rightarrow \Delta(S)$
5) C is a cost function. $C: S \times A \rightarrow \Re$
6) Z is an observation function. $Z: S \times A \rightarrow \Delta(Z)$

where $\Delta(\cdot)$ denotes the set of probability distributions. At time t+1, given $\{(s^k, o^k, a^k)\}_{k \le t}$, the system transits to the state s^{t+1} with probability $Prob(s^{t+1} \mid s^t, a^t)$, and the state then generates the observation o^{t+1} with probability $Prob(o^{t+1} \mid s^{t+1}, a^t)$. The states s^{t+1} are *not observable*. Actions are chosen with knowledge of the past observations, actions, and/or the distribution of the initial state. We consider reactive policies, mainly for their notational simplicity. A reactive policy is a randomized stationary policy such that the probability of taking an action is a function of the most recent observation only. The process $\{(s^t, o^t, a^t)\}$ jointly forms a Markov chain under a reactive policy, and so does the marginal process $\{(s^t, o^t)\}$, (marginalized over actions a^t).

The performance state of a system at time t is defined as a combination of delay and power dissipation values. The chip temperature (profile) at time t is the chip temperature measured and reported by a number of distributed on-chip temperature sensors. Note that the time units are abstractly defined and the task of casting them to absolute time units (micro or milli seconds) is achieved by the system developer.

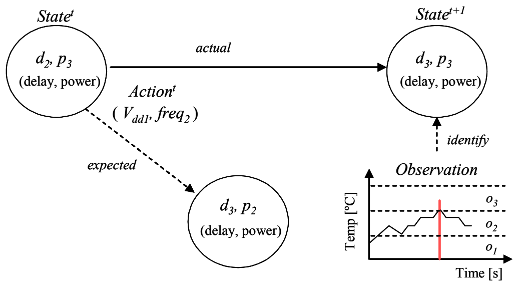

Figure 2. The rationale for developing the POSMDP framework.

POSMDPs are simply semi-Markov decision processes (SMDPs) with hidden states and observations generated by states. Figure 2 illustrates the key problem that the POSMDP framework addresses. In this figure, we define the chip temperature profile at time t to be one of the three observations: o_1, o_2, and o_3. Consider that, starting from a system state $s^t(d_2, p_3)$ at epoch time t, the power manager issues an action (V_{dd1}, $freq_2$) based on the accepted policy, and as a result, the system is expected to move into a new state s^{t+1} (d_3, p_2) at time t+ε. However, because of the PVT variations, etc., the resulting system state may actually be s^{t+1} (d_3, p_3) when it is observed at epoch time t+1. Thus, the power manager, which is conscious of PVT variations, makes decision for power management while estimating the system state based on information obtained from observations. The key contribution of our POSMDP framework is to recognize this uncertainty about the next state of the system, which is in turn caused by PVT variations.

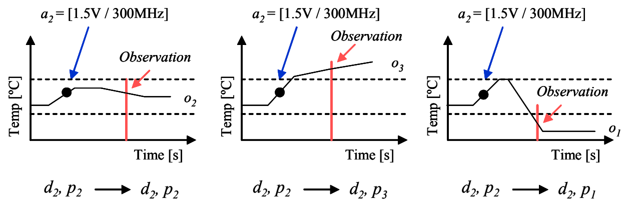

Figure 3. Example of three possible observations.

The power manager can choose an action from a finite set of actions, i.e., dynamic voltage and frequency scaling (DVFS) sets, at decision epochs. Although we know the issued action with certainty, the resulting state is not known in advance because of the PVT variations. For example, Figure 3 shows scenarios where starting in the active mode and with an action a_2 issued at time t, the next system state may be one of three possible ones as observed at time t+1, that is, the power manager cannot know for certain which next state will occur.

The state transition probability function determines the probability of a transition from a performance state s to another state s' after executing action a, i.e., $T(s', a, s) = Prob(s^{t+1} = s' \mid a^t = a, s^t = s)$.[1] Let $p_a(s, s', t)$ denote the probability that as a consequence of choosing action a when the system performance state is s, the state equals s' after time t. Note that $p_a(s, s', t)$ can be used to calculate the expected transition time between decision epochs. We consider a cost function which assigns a quantitative cost value to each state and action pair, i.e., we adopt the conventional approach whereby the expected cost is $k(s, a)$ incurred when action a is chosen in state s. The costs can be set by the applications or the developers.

An observation function, which captures the relationship between the actual performance state and the observation, may be defined as the probability of making observation o' after taking action a that should have landed the system in s', i.e., $Z(o', s', a) = Prob(o^{t+1} = o' \mid a^t = a, s^{t+1} = s')$.

3.2 Policy Representation in POSMDP

In partially observable environments, since a power manager cannot fully observe the underlying performance state of the system, it makes decisions based on the observable system history H. Note that the system history is a sequence of state and action pair such as $<s^0, a^0>$, $<s^1, a^1>$,…, $<s^t, a^t>$, making this a non-Markovian process. The power manager receives an observation o' which is dependent on s' and a. Although the observation gives the power manager some evidence about the current state s, s is not known exactly. Thus, we maintain a distribution over states called a belief state b [7]. The belief state for state s is denoted as $b(s)$, and the sum of belief states over all states is equal to 1, i.e., $\Sigma_{s\in S} b_s = 1$. Hence, by using the belief state space B, a properly updated probability distribution over the performance state S, we can convert the original POSMDP into a fully observable SMDP, so-called belief state SMDP [14]. A properly updated belief state, $b'(s')$, after action a and observation o', may be calculated from the previous belief state $b(s)$ as follows:

$$b'(s') = Prob(s' \mid o', a, b) = \frac{Z(o', s', a)\sum_{s\in S} T(s', a, s) b(s)}{Prob(o' \mid a, b)} \quad (1)$$

In this equation, the numerator consists of the observation function, transition function, and current belief state. The denominator is independent of s', and can be regarded as a normalization factor.

The power manager's goal is to choose a policy that minimizes a cost function, C. Basically, a particular policy tells the power manager what action to perform, and what to do next contingent on an observation. Let $\pi : B \to A$ represent a stationary policy that maps the probability distribution over belief states to actions. By incorporating the expectation over actions, the cost of a stationary policy π can be determined by using the Bellman equation [15] as

$$C^{\pi}(b) = \sum_{s\in S} b(s) k(s,a) + \gamma \sum_{o\in O} Prob(o' \mid a, b) C^{\pi}(b') \quad (2)$$

where γ is a discount factor, $0 \leq \gamma < 1$. Simply speaking, the power manager must execute the action a prescribed by policy π, and then update its probability distribution over the system's performance states according to equation (1). The optimal action to take at b is obtained by

$$\pi^{*}(b) = \arg\min_{a\in A} C^{\pi}(b) \quad (3)$$

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

A standard method of finding the optimal infinite policy π is to iterate cost function for POSMDP by using a sequence of optimal finite cost functions.

Figure 4 (a) shows a graphical *influence diagram* [14] of POSMDP, where the power manager generates an observation (caused by the action and state pair). Note that an arrow simply indicates an influence direction in the diagram. Now, we can convert it into the belief-state SMDP as shown in Figure 4 (b). The process of maintaining the belief state is Markovian, which means that the SMDP problem with the belief state can be solved by adapting the *value iteration* algorithm, which iterates on the optimal *value* of every state (defined as the expected infinite discounted sum of reward that the system will gain if it starts in that state and executes the optimal policy.) The optimal policy is then obtained by choosing, in every state, the action that minimizes the estimated discounted cost, using the current estimate of the value function. The value iteration technique may be contrasted with the *policy iteration* technique, which manipulates the policy directly, rather than finding it indirectly via the optimal value function. In practice, value iteration is much faster per iteration, but policy iteration takes fewer iterations. In our problem setup, we have found the value iteration to be more efficient than the policy iteration. Details of the value iteration technique for the SMDP is omitted to save space. Interested readers may refer to [7].

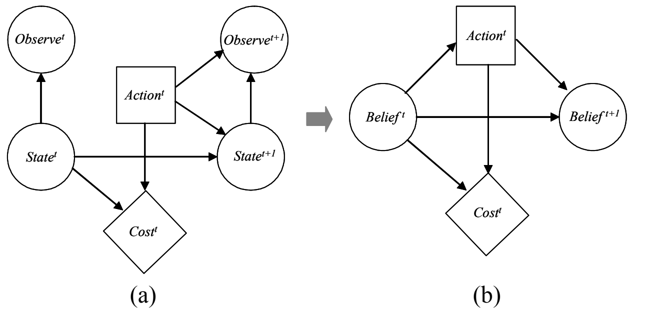

Figure 4. The joint influence diagram for a) POSMDP and b) belief-state SMDP.

4. Variability-aware Power Management

Considering the uncertainty in optimization problem, we present a variability-aware dynamic power management technique based on the POSMDP framework.

4.1 Power Management Formulation

Let a sequence of belief states $b^0, b^1, \ldots, b^n$ denote a processing path δ from b^0 to b^n of length n with the property that $p(b^0, b^1), \ldots, p(b^{n-1}, b^n) > 0$, where $p(x, y)$ is the probability that the system moves to state y from state x. For a policy π (cf. equation (3)), the discounted cost C of a processing path δ of length n is defined as

$$C^{\pi}(\delta) \triangleq \sum\nolimits_{i=0}^{n} \gamma^{t_i} cost(b^i, a^i) \qquad (4)$$

where exponent t_i denotes the duration of time that the system spends in belief state b^i before action a^i causes a transition to state b^{i+1}, and $cost(b, a)$ is the expected cost rate, given by

$$cost(b,a) = pow(b) + \frac{1}{\tau(b,a)} \sum_{b' \in B} Prob(b' \mid b,a) ene(b,b') \qquad (5)$$

where $pow(b)$ is the power consumption of the system in belief state b, $Prob(b' \mid b, a)$ is the probability of being in belief state b' after action a in state b, $ene(b, b')$ is the energy required by the system to transit from state b to b', and $\tau(b, a)$ is the expected duration of time that the system spent in state b if action a is chosen. Thus, we can compute the expected power consumption of the system in active mode, given that the system starts in state b, as $actpow^{\pi}_{avg}(b) = EXP[C^{\pi}(\delta)]$. Considering the expectation with respect to π over the set of processing paths starting in state b, the total energy dissipation of the system is defined as:

$$ene_{total} = actpow^{\pi}_{avg}(b) \cdot \sum_{l \in L} exe_{l.\pi} + slepow_{Vdd.Vth} \cdot (T_d - \sum_{l \in L} exe_{l.\pi}) \qquad (6)$$

where $slepow_{Vdd.Vth}$ is the power consumption of the system in the sleep mode, $exe_{l.\pi}$ is the execution time of task l, where $l \in L$ (set of tasks), running under policy π, and T_d is the given deadline. Changing the voltage level (and correspondingly the operating frequency) of the system affects the execution time of the tasks as follows. Let V_i denote the operating voltage and f_i the clock frequency of the system, which are set by the power manager. For the simplicity of discussion, we define the workload of task l, $N_{i,l}$, as the number of clock cycles required to complete task l at operating voltage V_i. Let variable $x(l, i)$ represent the percentage of the workload of task l running at voltage V_i under policy π. The execution time of task l under policy π may be calculated as

$$exe_{l,\pi} = \sum\nolimits_{i=1}^{m} x(l,i) \cdot N_{i,l} \Big/ f_i \qquad (7)$$

where m is the number of optimal voltage settings, and $\Sigma_i x(l,i) = 1$. Thus, we can calculate the remaining idle time as $D_l - exe_{l,\pi}$, which is used for slack calculation to achieve energy savings, where D_l is the deadline of task l.

A block diagram for the proposed variability-aware power management solution is provided in Figure 5. We assume that the system has two modes: active and sleep modes for simplicity (the idle mode is regarded as being a part of the active mode.) In the active mode, the system can switch between different speed levels, i.e., power-saving states, as commanded by the power manager

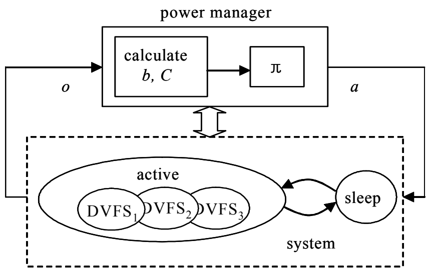

Figure 5. The diagram of variability-aware power management.

4.2 Optimal Policy Algorithms

Considering the optimal policy during power management, we introduce two algorithms that incorporate our proposed framework. The first algorithm is an offline algorithm based on SMDP modeling, which finds the optimal policy but assumes that the state transition probabilities are known with certainty. The second algorithm is an online version that could cope with the unknown state of the system and utilizes the POSMDP framework.

As for the offline algorithm, if the state transition probabilities $T(s', a, s)$ are known, the SMDP-based algorithm in determining optimal policy can be described as a linear programming problem as shown in [13]:

$$\begin{aligned}
\min\ & \sum_{s}\sum_{a} actpow^{\pi}_{\text{avg}}(s)\varphi(s,a) \\
\text{s.t.}\ & \sum_{a}\varphi(s,a) - \sum_{s'}\sum_{a}\varphi(s',a)T(s',a,s) = 0 \\
& \sum_{s}\sum_{a}\varphi(s,a)\tau(s,a) = 1 \\
& \varphi(s,a) \geq 0 \quad all\ s \in S, a \in A
\end{aligned} \tag{8}$$

where $actpow^{\pi}_{avg}(s)$ is the expected power in state s, and $\varphi(s,a)$ is the frequency that the system is in state s and action a is issued. Note that the optimal solution to the SMDP policy optimization problem belongs to the set of deterministic policies [7].

Figure 6 describes the proposed online algorithm for variability-aware power management when the performance state is not known beforehand. The algorithm includes an operation to estimate the next step as $s^{t+1} = \boldsymbol{F}s^t$, where $\boldsymbol{F}$ is a projection matrix that has an induced transition probability F_{ij}, a weighted sum over actions of transition probabilities, $T(s_j, a_k, s_i)$, with conditional probability of action a in state s, $Prob(a_k \mid s_i)$, given by

$$F_{ij} = \sum_{k} T(s_j, a_k, s_i) Prob(a_k \mid s_i) \tag{9}$$

The system history is a sequence of state, observation, and action triples such as $<b^0, o^0, a^0>, <b^1, o^1, a^1>, \ldots, <b^t, o^t, a^t>$ and the cost is provided by the users e.g., in a table-lookup. With n states and m actions, this algorithm takes $O(n)$ and $O(m)$ times for finding a belief state and an optimal action, respectively, which results in total $O(nm)$ running time.

```
n: the number of states
m: the number of actions
input: o, l, D_l
output: π
 1: observe temperature o
 2: estimate the next state,  s' = Fs
 3:      for x = 1 to n
 4:           if (o_x = o)
 5:           calculate belief state b
 6:                for y = 1 to m
 7:                     calculate D_l – exe_{l,ay} (slack)
 8:                     find a_y s.t. minimum ene
 9: update system history  H
10: return π
```

Figure 6. Online algorithm for variability-aware DPM.

5. Experimental Results

In the experimental setup, we implemented a 32bit RISC processor compatible with [17] in 130nm CMOS technology, which has 3 operating voltage levels and dual threshold voltages. We developed the proposed algorithm in Matlab, which allows us to rapidly consider multiple scenarios with respect to the magnitude and distribution of PVT variations. We obtained SAIF (Switching Activity Interchange File) [18] by back-annotated RTL simulation, and then executed the Power Compiler [18] to achieve accurate power value.

We first analyze the performance behavior of the RISC processor as a function of the process variations. Figure 7 shows a set of power-delay curves for the RISC processor, obtained by running the Power Compiler for various process conditions and V_{th} variations. For example, if we change the degree of variation for V_{th}, then the design in the SS condition (i.e., 1.35 V_{dd}, 125°C) can result in leakage power dissipation between 1.5uW and 6.2uW. The dynamic power dissipation is, however, independent of the V_{th} variations and since in our design and technology node, the dynamic power dissipation constitutes the bulk of the total power dissipation, the variation in total power with respect to V_{th} is rather small. In our experiment, we set the 3-σ variations in the operating frequency and supply voltage level of the chip to be 10%. This is a reasonable assumption for a chip realized in the 130nm CMOS technology.

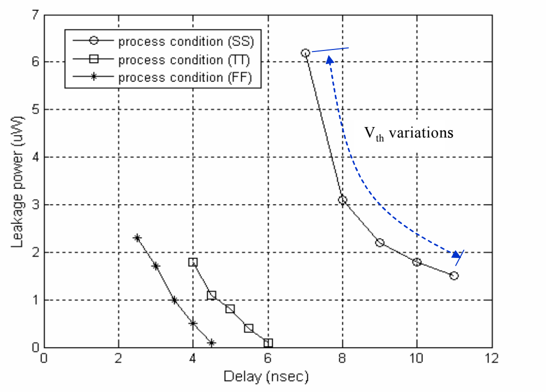

Figure 7. Power-delay curves for different levels of variability.

The second experiment is to demonstrate the effectiveness of the proposed POSMDP framework and power management technique. First, we set the parameter values for the evaluation of the POSMDP framework as shown in Table 1 and Figure 8. The performance state (s_1, s_2, and s_3) in Figure 8 is defined as a combination of power dissipation and execution delay for the RISC processor.

Table 1. Parameter values for a given experiment.

cost	Action	Description	Observation	Description
[$k(s_1, a)$ $k(s_2, a)$ $k(s_3, a)$]				
[1.5 1 0]	a_1	[1.35V / 200MHz]	o_1	[50°C ≤ temp < 65°C]
[1 0 1]	a_2	[1.50V / 350MHz]	o_2	[65°C ≤ temp < 75°C]
[0 1 1.5]	a_3	[1.65V / 500MHz]	o_3	[75°C ≤ temp < 90°C]

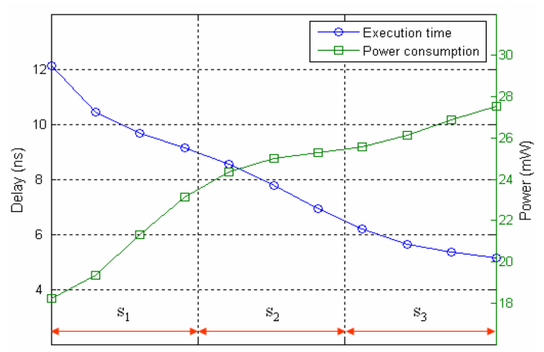

Figure 8. Performance states for a given experiment.

Second, we arbitrarily choose a sequence of 20 application programs, which include SPECint2000 *gcc*, *gap*, and *gzip* benchmarks e.g., gap_1-$gzip_2$-gap_3-gcc_4-…-gap_{20}, where $program_i$ is the *i*-th program in the sequence. Next, the sequence of programs is executed on the processor to calculate the belief states based on the estimated temperature which serves as the observation, where we use thermal parameters extracted from the commercial data sheet for a QFP package as adopted in [9]. We assume that the processor starts from $<s_1, s_2, s_3> = <1, 0, 0>$ as the state point, as shown in Figure 9. Then, the belief states are evaluated based on action and observation, as the processor executes the sequence of programs as in Figure 9, where we use the offline algorithm in determining the policy.

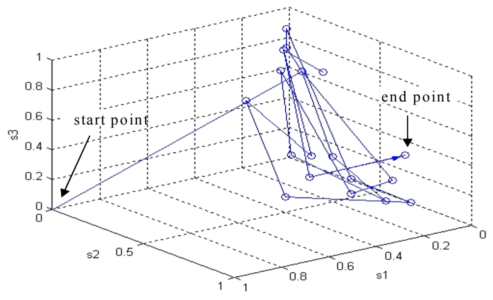

Figure 9. Trace of belief states based on the temperature observation.

Simulation results in Figure 10 demonstrate that the proposed variability-aware power management technique ensures energy savings in the presence of large within-chip variations. We use the energy-delay-product (which is inversely proportional to MIPS^2/watt) [12] as the figure of merit. Both cases (a) and (b) in Figure 10 show the power consumption, obtained by running the sequence of 20 programs, as the worst-case (more energy) and best-case (less energy), respectively, without considering the PVT variation effects. The conventional power management methods (which are unaware of the PVT variations) can produce 100% difference in energy-delay-product (EDP) figure. Our proposed method, case (c), achieves energy savings while considering the variations (i.e., we set various values for temperature, V_{th}, and V_{dd} during simulation), resulting in EDP of (normalized) 1.27 for the online algorithm. The offline algorithm results in (normalized) 1.09, similar to the abovementioned best-case without considering the variability issue. Table 2 summarizes these simulation results in terms of power dissipation, energy, and energy-delay-product.

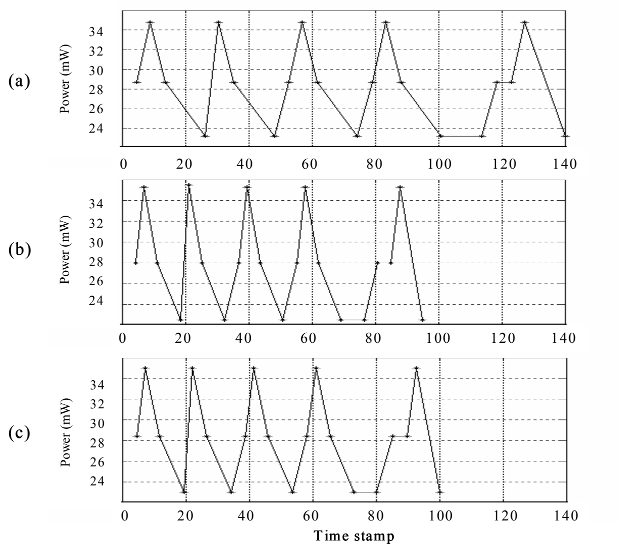

Figure 10. Results on various power management: (a) Worst-case (b) Best-case (c) The proposed method (online).

Table 2. Summary of Figure 10 (power, delay per program).

	Description	Min. power / delay	Max. power / delay	Energy (normalized)	EDP (normalized)
Case (a)	-Worst-case -Assume No-variation	23.2mW / 12.11ns	34.8mW / 4.29ns	1.42	2.06
Case (b)	-Best-case -Assume No-variation	23.0mW / 7.18ns	35.3mW / 3.61ns	1.00	1.00
Case (c) (online)	-Variability-Aware	23.1mW / 7.66ns	35.0mW / 3.73ns	1.15	1.27

6. Conclusion

We proposed a variability-aware dynamic power management technique which brings the variational effects to the forefront of power management framework. The proposed uncertainty management framework, which is based on POSMDP, controls the uncertain states of the system and makes a decision (voltage and frequency setting) to achieve energy savings. Being able to predict the variational effect would allow significant reduction of the uncertain behavior of the system, improving the DPM robustness.

References

[1] 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.

[2] H. Su, F. Liu, A. Devgan, E. Acar, and S. Nassif, "Full Chip Leakage Estimation Considering Power Supply and Temperature Variations," *Proc. of ISLPED*, Aug., 2003.

[3] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Kehavarzi, and V. De, "Parameter Variations and Impact on Circuits and Microarchitecture," *Proc. of DAC*, June, 2003.

[4] M. Lie, W.S. Wang, and M. Orshansky, "Leakage Power Reduction by Dual-Vth Designs Under Probabilistic Analysis of Vth Variation," *Proc. of ISLPED*, Aug., 2004.

[5] A. Basu, et al., "Simultaneous Optimization of Supply and Threshold Voltages for Low-Power and High-Performance Circuits in the Leakage Dominant Era," *Proc. of DAC*, June, 2004.

[6] K. Keutzer, and M. Orshansky, "From Blind Certainty to Informed Uncertainty," *Proc. of Int'l Workshop on Timing Issues*, Dec., 2002.

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

[8] A. H. Ajami, K. Banerjee, and M. Pedram, "Scaling Analysis of On-chip Power Grid Voltage Variations in Namometer Scale ULSI," *Journal of Analog Integrated Circuits and Signal Processing*, Vol. 42, No. 2, pp. 277-290, Springer, 2005.

[9] Y. Cheng, C. Tsai, C. Teng, and S. Kang, *Electrothermal Analysis of VLSI Systems*. Kluwer Academic Publishers, 2000.

[10] E. Humenay, et al, "Toward an Architectural Treatment of Parameter Variations," *Univ. of Virginia, Tech. report CS-2005-16*, Sep. 2005.

[11] K. Skadron, et al., "Temperature-Aware Microarchitecture," *Proc. of Int'l Symposium on Computer Architecture*, June 2003.

[12] M. R. Stan, and K. Skadron, "Power-Aware Computing," *Journal of IEEE Computer*, Vol. 36, No. 1, Jan. 2003.

[13] 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, Oct. 2001.

[14] A.R. Cassandra, et al., "Acting Optimally in Partially Observable Stochastic Domains," *Proc. of 12th Conference on AI*, Aug. 1996.

[15] R.E. Bellman, *Dynamic Programming*. Princeton University Press, Princeton, 1957.

[16] D. Brooks and M. Martonosi, "Dynamic Thermal Management for High Performance Microprocessor," *Proc. of HPCA*, Jan. 2001.

[17] http://www.opencores.org. OpenRISC 1000 processor.

[18] http://www.synopsys.com. Synopsys Power Compiler Documents.