On-Line Reliability-Aware Dynamic Power Management for Real-Time Systems

Ming Fan[1], Qiushi Han[2], Shuo Liu[2] and Gang Quan[2]
[1] Broadcom Corporation, 3151 Zanker Road, San Jose, CA 95134, USA
[2] Florida International University, Miami, FL 33174, USA
E-mail: mingfan@broadcom.com, {qhan001,sliu005,gaquan}@fiu.edu

Abstract— As VLSI technology continues to advance, the aggressive scaling of transistor size not only dramatically increases the power consumption of a processor chip, but also significantly decreases the reliability of the chip. While Dynamic Voltage and Frequency Scaling (DVFS) techniques have been shown very effective in power reduction, they can adversely affect the reliability and stability of computing systems. In this paper, we present an online power management approach to schedule real-time tasks on a single processor platform to minimize the energy consumption without compromising the overall system reliability. The proposed algorithm recycles the reserved redundant resources dynamically and thus can reduce the energy consumption more effectively. As shown in our simulation study, by exploiting the run-time dynamics, we proposed technique can significantly outperform the previous work in energy savings.

Keywords— Dynamic Power Management, Reliability, Real-Time

I. Introduction

Nowadays, embedded real-time systems grow rapidly in both scale and complexity thanks to the advancement in microprocessor technologies. One key problem, as a result of the increasing number of transistors on a single die and the aggressive scaling in the transistor size, is the dramatically increased power consumption, which in turn poses severe constraints on the operations of a system. Dynamic voltage and frequency scaling (DVFS) is one of the most commonly used technique for power/energy management, which has been well studied in [1], [2]. With DVFS enabled processors, the supply voltage and frequency are lowered at run-time to achieve energy savings. However, as shown in recent studies [3], [4], DVFS can directly and adversely affect the system reliability.

As transistors become smaller and smaller, the reliability problem for a processor becomes more and more significant. In addition, reducing the voltage and operating frequency of a processor exacerbates the reliability problem. For example, it is reported that the transient fault rate occurred in a processor usually increases in several orders of magnitude under low power/frequency condition. Transient fault refers to the temporary malfunction of a processor, usually caused by electromagnetic interferences or cosmetic ray radiations, that can lead to temporary errors in computation and corruptions in data [5], [6], [7]. With the increased complexities in both system architecture and applications, the reliability issue is becoming more and more critical in the modern real-time embedded system design.

It is a well-known fact that, in real-time systems, there are usually a large difference between the worst case execution time and best case execution time for the same real-time task. Therefore the approach that can take advantage of the run-time dynamics can be very effective in saving energy. As a result, we want to study how to employ on-line scheduling techniques to save energy without degrading the system reliability.

In this paper, we present an on-line reliability-aware dynamic power management approach to schedule frame-based task set on single processor platform. Compared with the existing work, we have made a number of novel contributions:

- First, we made an interesting observation that the system reliability varies with the executions of real-time tasks. By taking the system on-line property into consideration, instead of guaranteeing the system original reliability through off-line approach, we satisfy the reliability requirement through an on-line approach.
- Secondly, by recycling the preserved computing resource dynamically, our proposed algorithm can effectively exploit the run-time slacks to adjust the frequencies of real-time tasks such that the system energy consumption can be minimized without compromising the system reliability.

The rest of the paper is organized as follows. Section II describes system models and other background information necessary for this paper. Section III present our proposed reliability-aware dynamic power management algorithm. Experiments and results are discussed in Section IV, and we present the conclusions in Section V.

II. Preliminary

In this section, we introduce our real-time system model, fault model, reliability model, and power/energy models, respectively.

A. *The real-time system model*

The real-time system applications considered in this paper consists of N independent tasks, denoted as $\Gamma = \{\tau_1, \tau_2, ..., \tau_N\}$. All tasks in Γ have the same deadline D, but with different execution requirements. We denote the execution time of τ_i as c_i. For the rest of this paper, we assume that

Γ is sorted with increasing execution time.

We consider the uniprocessor system with DVFS operation capability. Assume that the available discrete frequencies can vary from the minimum value f_{min} to the maximum value f_{max}, and all frequency values are normalized with respect to f_{max}, (i.e. $f_{max} = 1$). Specifically, let $\mathcal{F}$ denote the discrete frequency set which consisting of L different frequencies, i.e. $\mathcal{F} = \{f_1, f_2, ..., f_L\}$, where $f_i < f_j$ if $1 \leq i < j \leq L$. The execution time of τ_i under the frequency f_i is given by c_i/f_i.

B. The transient fault model

During the execution of an application, faults may occur due to various reasons, such as hardware failures, software errors and the effects of electromagnetic interference and cosmic ray radiations. Previous research has shown that transient faults occur much more frequently than permanent faults [8]. Therefore, in this work, we focus on transient faults and adopt the backward recovery technique to tolerate such faults [9].

More specifically, transient faults are assumed to follow Poisson distribution with an average arrival rate λ [10]. Considering the effects of voltage scaling on transient faults, the average fault rate λ depends on system supply voltage and processing frequency. In this paper, we use the exponential fault rate model used in [11], [12], [3], [4]:

$$\lambda(f) = \lambda_0 \cdot 10^{\frac{d\cdot(1-f)}{1-f_{min}}} \tag{1}$$

where the exponent $d(>0)$ is a constant, indicating the sensitivity of fault rates to voltage scaling. λ_0 is the average fault rate corresponding to the maximum frequency f_{max}. That is, reducing the frequency for energy savings will result in exponentially increased fault rates. The maximum average fault rate is assumed to be $\lambda_{max} = \lambda_0 \cdot 10^d$, which corresponds to the lowest frequency f_{min}.

C. The reliability model

Given a frequency assignment for N tasks, i.e. $F = \{f_1, f_2, ..., f_N\}$, we introduce four concepts, i.e. the *task reliability*, the *optimal system reliability*, the *static system reliability* and the *dynamic system reliability* to model and describe the reliability for task and system, respectively.

Definition 1: Given a task τ_i, the *task reliability* of τ_i under frequency f_i, denoted as $R_i(f_i)$, is defined as [3], [4].

$$R_i(f_i) = e^{-\lambda(f_i)\cdot\frac{c_i}{f_i}} \tag{2}$$

where $\lambda(f_i)$ is given by Equation (1).
$R_i(f_i)$ represents the probability of completing τ_i's execution successfully under frequency f_i with no fault occurrence.

Definition 2: Given a frame-based task set Γ consisting of N tasks, the *optimal system reliability* of Γ, denoted as R_o, is computed as [3], [4].

$$R_o = \prod_{i=1}^{N} R_i(f_{max}) \tag{3}$$

where f_{max} is maximum available frequency in the system.
R_o represents the probability of completing all tasks successfully at the maximum available frequency f_{max} with no fault occurrence.

Definition 3: Given a frame-based task set $\Gamma = \{\tau_1, \tau_2, ...\tau_N\}$, a frequency assignment $F = \{f_1, f_2, ..., f_N\}$ and backup block assignment $B = \{b_1, b_2, ..., b_K\}$. The *static system reliability*, denoted as R_s, is defined as:

$$\begin{aligned} R_s^K(f_1, ..., f_N) &= R_1(f_1)\cdot R_s^K(f_2, ..., f_N) + \\ &\quad (1 - R_1(f_1)\cdot R_1(f_{max})\cdot R_s^{K-1}(f_2, ..., f_N) \end{aligned} \tag{4}$$

where $R_s^0(f_j, ..., f_N) = \prod_{i=j}^{N} R_i(f_i)$
R_s depicts the probability of completing all tasks under frequency assignment F with K-recovery blocks in B. The first term in equation (4) represents that task τ_1 completes successfully and all K recovery blocks are made available to the remaining tasks. Similarly, the second term represents that task τ_1 fails and subsequently is recovered with one recovery block. The remaining tasks will have (K-1) recoveries to complete their executions. According to work [11], the complexity of calculating $R_s^K(f_1, .., f_N)$ is $O(K\cdot N)$.

It is important to point out that when scheduling a task set Γ on-line, at certain time point t, the status of completed tasks are deterministic. Let $X(t)$ be an indicator vector for a task set Γ at time instance t, where $X(t) = \{x_1(t), x_2(t), ..., x_N(t)\}$, then $X(t)$ must satisfy the constraints below:

$$\begin{cases} x_i(t) = 0 & \text{if } \tau_i \text{ finished successfully at } t \\ x_i(t) = 1 & \text{if } \tau_i \text{ finished with fault at } t \\ \sum_{i=1}^{N} x_i(t) \leq K \end{cases} \tag{5}$$

Based on the notation of $X(t)$, we introduce the concept of *dynamic system reliability* as below.

Definition 4: Given a frame-based task set $\Gamma = \{\tau_1, \tau_2, ...\tau_N\}$, a frequency assignment $F = \{f_1, f_2, ..., f_N\}$ and backup block assignment $B = \{b_1, b_2, ..., b_K\}$. Let τ_j be the last completed task at certain time point t, then the *dynamic system reliability* at t, denoted as $R_d(t)$, is defined as:

$$R_d(t) = [\prod_{i=1}^{j}((1-x_i) + x_i\cdot R_i(f_i))]\cdot R_s^{K_d(t)}(f_{j+1}, ..., f_N) \tag{6}$$

where $K_d(t) = K - \sum_{i=1}^{j} x_i(t)$, and K is the number of total reserved recovery blocks.

Note that the first part on the right side of equation (6) (the production part by $\prod$) represents the deterministic reliability combing from the completed tasks, while the second part on the right side of equation (6) represents the expected reliability coming from the uncompleted tasks. $K_d(t)$ denotes the total left recovery blocks at time t that can be used by the uncompleted tasks, which is obtained by subtracting the consumed recovery block from the total available recovery block.

D. Power and energy model

The system power model considered in this paper is distinguished into two parts: frequency-independent part and frequency-dependent part[11], [4].

$$P = P_{ind} + C_{ef} f^{\alpha} \tag{7}$$

P_{ind} is the frequency-independent power, denoting the power consumed by off-chip devices such as main memory and external devices. $C_{ef} f^{\alpha}$ is the frequency-dependent power, including the CPU power [13].

C_{ef}, f and α are the effective switching capacitance, processing frequency and dynamic power exponent, respectively. Note that α is system-dependent constant, and in general is no smaller than 2. Thus, the energy consumption of a task τ_i running at the frequency f_i can be formulated as [4]:

$$E_i(f_i) = (P_{ind} + C_{ef} f_i^{\alpha}) \cdot \frac{c_i}{f_i} \tag{8}$$

Finally, the total system energy consumption under frequency assignment $F = \{f_1, f_2, ..., f_N\}$ with no fault occurrence is obtained by

$$E(f_1, f_2, ..., f_N) = \sum_{i=1}^{N} E_i(f_i); \tag{9}$$

In the next section, we introduce an on-line reliability-aware dynamic power management approach that minimizes the overall system energy consumption meanwhile maintaining the system feasibility and desired reliability.

III. Reliability-aware dynamic power management algorithm

The *Reliability-Aware Dynamic Power Management (RA-DPM)* algorithm is an on-line power management algorithm that dynamically adjusts the tasks' frequencies by utilizing potential redundant recovery blocks on-line. The system original reliability obtained by equation (3) can be guaranteed, and the original recovery requirement can be satisfied. To this end, we first introduce our *RA-DPM* algorithm, then discuss the its reliability property.

A. Algorithm details

RA-DPM algorithm dynamically adjusts the tasks' frequencies by utilizing run-time slacks which may be increment through recycling redundant recovery blocks. The purpose is to minimize the system energy consumption under the constraints of reliability and fault tolerance. Algorithm 1 shows the salient aspects of the RA-DPM.

The brief description of Algorithm 1 is below. Given a input task set Γ and discrete available frequency set Φ, we first initialize the all tasks' frequencies and recovery blocks by IRCS algorithm [11] (line 1). Note that, any algorithms that statically assign frequencies and recovery blocks is acceptable for our RA-DPM algorithm. Then calculate the system original reliability under f_{max} by equation (3) (line 2). And with ignoring fault occurrence, we can also calculate the system energy consumption under F by equation (9) (line 3). The task list $TList$ contains all un-scheduled tasks, and initialized to the input task set Γ. The "for loop" (from line 5 to line 26) schedules all tasks in $TList$ one by one with default order. If the number of left tasks in $TList$ is less than the number of recovery blocks, then we will recycle one certain redundant recovery block (from line 6 to line 20). In order to keep the original recover capacity, the maximum recovery block or the minimum recovery block will be recycled (from line 8 to line 16). If the maximum execution time among all un-scheduled tasks is less than the maximum recovery block capacity, then we can recycle that maximum reserved block. Otherwise, the minimum recover block will be recycled. Consequently, the new slacks

Algorithm 1 On-line RA-DPM algorithm

```
Require:
    1) Task set : $\Gamma = \{\tau_1, \tau_2, ...\tau_N\}$;
    2) Discrete frequencies : $\Phi = \{\phi_1, \phi_2, ..., \phi_L\}$,
       where $\phi_i \in [f_{min}, f_{max}], i = 1, 2, ..., L$;
1:  Initialize all tasks' frequencies ($F = \{f_1, f_2, ..., f_N\}$, $f_i \in \Phi$) and recovery blocks ($B = \{b_1, b_2, ..., b_K\}$) by IRCS algorithm [11];
2:  Compute the original reliability $R_o$ by equation (3);
3:  Compute the energy consumption $E_o$ by equation (9);
4:  $TList = \Gamma$;
5:  for $i = 1$ to $N$ do
6:  $B$ = current available recovery block set;
7:  $isRecycle = False$;
8:  if $|TList| < |B|$ then
9:  $c_{max} = max(\{c_j^{f_{max}} | \tau_j \in Q\})$;
10: $b_{min} = min(\{b_j | b_j \in B\})$;
11: $b_{max} = max(\{b_j | b_j \in B\})$;
12: if $c_{max} < b_{max}$ then
13: $b^* = b_{max}$;
14: else
15: $b^* = b_{min}$;
16: end if
17: $S = D - \sum_{i=1}^{N} \frac{c_i}{f_i} + b^*$;
18: Remove $b^*$ from $B$,;
19: $isRecycle = True$;
20: end if
21: if $isRecycle = True$ then
22: Re-assign frequencies of all tasks in $TList$ with respect of new slack time $S$, such that
    1) system on-line reliability($Q$) is no smaller than $R_o$,
    2) system energy consumption is no greater $E_o$;
23: end if
24: Schedule $\tau_i$ under frequency $f_i$ with actual fault case;
25: Remove $\tau_i$ from $TList$;
26: end for
```

can be calculated (line 17). By applying the new slacks to further minimize energy consumption, the reliability constraint must be satisfied (line 22). Current task τ_i will be scheduled with its corresponding frequency, and the actual fault status will be taken into account as well. Finally, remove the current task τ_i from $TList$ when it is finished.

B. Reliability analysis

After introducing our proposed RA-DPM algorithm, we discuss the task reliability and system reliability under the frequencies determined by RA-DPM.

Let Q_i represent the actual reliability value of τ_i, then we show that RA-DPM can guarantee the original reliability of τ_i.

Lemma 1: Let Q_i represent the reliability of τ_i after scheduled by RA-DPM, then we have

$$Q_i \geq R_i(f_{max}) \tag{10}$$

Proof: The proof is

case 1: τ_i is completed successfully

$$Q_i = 1 \geq R_i(f_{max}) \tag{11}$$

case 2: τ_i is completed with fault occurrence, and consumes on backup block.

$$\begin{aligned} Q_i &= R_i(f_i) + (1 - R_i(f_i)) \cdot R_i(f_{max}) \\ &= R_i(f_{max}) + (1 - R_i(f_{max})) \cdot R_i(f_i) \\ &\geq R_i(f_{max}) \end{aligned} \tag{12}$$

From equation (11) and equation (12), we can directly conclude the result shown in equation (10). □

Then, we discuss the system reliability under DVFS.

Lemma 2: Let $\{f_1, ..., f_N\}$ be the frequency assignment result and K be the backup block number by RA-DPM, then we have

$$R_s^K(f_1, ..., f_N) \geq R_o \tag{13}$$

Proof: Given a frequency assignment $\{f_1, f_2, ..., f_N\}$, according to equation (4), the system reliability can be calculated by

$$\begin{aligned} R_s^K(f_1, ..., f_N) = \; & R_1(f_1) \cdot R_s^K(f_2, ..., f_N) + \\ & (1 - R_1(f_1)) \cdot R_1(f_{max}) \cdot R_s^{K-1}(f_2, ..., f_N) \end{aligned}$$

Moreover, since $R_s^K(f_2, ..., f_N) \geq R_s^{K-1}(f_2, ..., f_N)$the above equation can be rewritten as

$$\begin{aligned} R_s^K(f_1, ..., f_N) \geq (R_1(f_{max}) \quad + \quad & (1 - R_1(f_{max})) \cdot R_1(f_1)) \cdot \\ & R_s^{K-1}(f_2, ..., f_N) \end{aligned}$$

Since $R_1(f_{max}) + (1 - R_1(f_{max}) \geq R_1(f_{max})$, from the above we can further derive that

$$R_s^K(f_1, ..., f_N) \geq R_1(f_{max}) \cdot R_s^{K-1}(f_2, ..., f_N) \tag{14}$$

Expand the last part on the right side of the above, and simplify the result following the same rule, we can derive that

$$R_s^K(f_1, ..., f_N) \geq R_1(f_{max}) \cdot R_2(f_{max}) ... R_N(f_{max}) \tag{15}$$

According to equation (3), the right side of the above can be replaced by R_o, then we can obtain

$$R_s^K(f_1, ..., f_N) \geq R_o$$

□

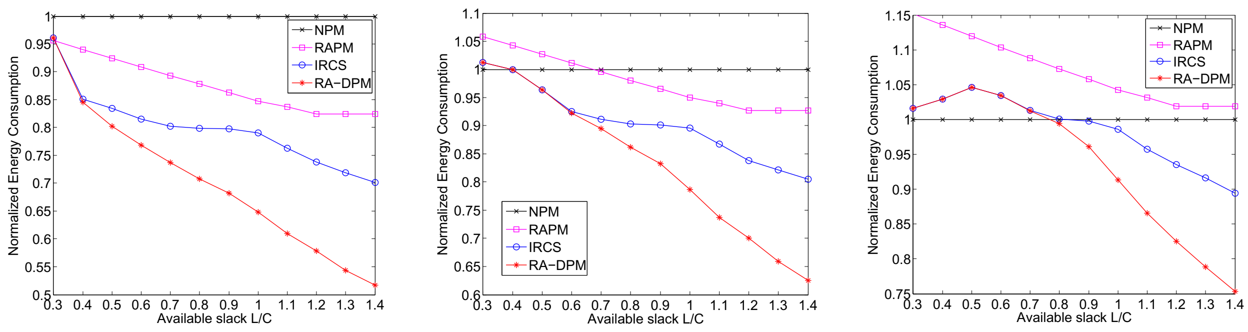
IV. Experimental evaluation

In this section, we investigate the performance of our proposed algorithms with experimental simulations. We compare our *RA-DPM* with other three approaches, i.e. *No Power Management (NPM)* approach (baseline, which assigns maximum frequency to all task, *IRCS* approach [11] and the *RAPM* approach [3].

In the simulations, transient faults are assumed to follow Poisson distribution with an average fault rate of $\lambda_0 = 10^{-6}$ at f_{max}, which is a realistic fault rate as reported in [14]. Moreover, the fault rate exponent d is set to 2. The six discrete frequency levels that we assume are modeled according to Intel Xscale processor [15]. We use a cubic frequency-dependent power component P_d which is equal to unity at $f_{max} = 1.0$.

We generate 1000 different task sets, each of which consists of 10 tasks. For each task, i.e. τ_i, its worst-case execution time c_i is randomly generated through a uniform distribution in the range of $[1ms, 10ms]$. In our experiments, we set the threshold of the system reliability to $R_g = \rho_0 = \prod R_i(f_{max})$, which is the system reliability when all tasks run at the maximum frequency f_{max}. P_{ind} is set to 0.05 as a constant and the available slack is denoted as $L = D - C$, where D is the deadline of the application and $C = \Sigma c_i$. Then the slack-to-execution ratio is calculated by $\frac{L}{C}$. All energy consumption results are normalized with respect to the baseline approach, i.e. *NPM*. The results of different approaches are collected and plotted in Figure 1.

Figure 1 shows the results of energy consumption of four approaches under three different fault scenarios, i.e. fault-free, 1-fault, and 2-fault, respectively. As illustrated in Figure 1(a), all three methods can achieve energy savings compared to *NPM*. Clearly, our *RA-DPM* outperforms the other two techniques. For instance, when the slack to execution ratio is 1, our approach attains approximately 18% more energy saving than *IRCS* and 24% more than *RAPM*. The experiment is repeated with 1-fault occurrence in Figure 1(b). The results follow the same trend, however has its own characteristics. Unlike the previous experiment, *NPM* consumes the least amount of energy when the slack-to-execution ratio is relatively small. This scenario is caused by the re-execution of one faulty task at maximum frequency and subsequent lack of space for DVFS. With the increasing slack time, our approach starts to show its advantage and achieve high energy saving. This situation is further illustrated in 1(c), our approach still outperforms the others by a high amount.



(a)Fault number is 0 (b)Fault number is 1 (c)Fault number is 2

Fig. 1. Slack impact on energy consumption under different fault occurrence

V. Conclusions

In this paper, we presented and evaluated a novel reliability aware power management algorithm, called *RA-DPM*, which aims to minimize energy consumption while keeping the system's reliability at a desired level. *RA-DPM* dynamically adjusts the tasks' frequencies by utilizing run-time slacks which may be increased through recycling redundant recovery blocks. It differs from the existing works where task frequencies assignments are predetermined, therefore it is more flexible and adaptive. The experimental results demonstrate that the proposed algorithm can significantly improve the energy savings compared with the previous works.

References

[1] R. Jejurikar and R. Gupta, "Dynamic voltage scaling for systemwide energy minimization in real-time embedded systems," in *Low Power Electronics and Design, 2004. ISLPED '04. Proceedings of the 2004 International Symposium on*, aug. 2004, pp. 78 –81.

[2] G. Quan and X. Hu, "Energy efficient fixed-priority scheduling for real-time systems on variable voltage processors," in *Design Automation Conference, 2001. Proceedings*, 2001, pp. 828 – 833.

[3] D. Zhu and H. Aydin, "Energy management for real-time embedded systems with reliability requirements," in *Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on*, nov. 2006, pp. 528 –534.

[4] D. Zhu, R. Melhem, and D. Mosse, "The effects of energy management on reliability in real-time embedded systems," in *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*, ser. ICCAD '04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 35–40. [Online]. Available: http://dx.doi.org/10.1109/ICCAD.2004.1382539

[5] J. Srinivasan, S. Adve, P. Bose, and J. Rivers, "The impact of technology scaling on lifetime reliability," in *Dependable Systems and Networks, 2004 International Conference on*, june-1 july 2004, pp. 177 – 186.

[6] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*, 2002, pp. 389 – 398.

[7] D. Ernst, S. Das, S. Lee, D. Blaauw, T. Austin, T. Mudge, N. S. Kim, and K. Flautner, "Razor: circuit-level correction of timing errors for low-power operation," *Micro, IEEE*, vol. 24, no. 6, pp. 10 –20, nov.-dec. 2004.

[8] X. Castillo, S. R. McConnel, and D. P. Siewiorek, "Derivation and calibration of a transient error reliability model," *IEEE Trans. Comput.*, vol. 31, pp. 658–671, July 1982. [Online]. Available: http://dx.doi.org/10.1109/TC.1982.1676063

[9] D. K. Pradhan, Ed., *Fault-tolerant computer system design*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1996.

[10] Y. Zhang and K. Chakrabarty, "Energy-aware adaptive checkpointing in embedded real-time systems," in *Proceedings of the conference on Design, Automation and Test in Europe - Volume 1*, ser. DATE '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 10 918–. [Online]. Available: http://dl.acm.org/citation.cfm?id=789083.1022841

[11] B. Zhao, H. Aydin, and D. Zhu, "Generalized reliability-oriented energy management for real-time embedded applications," in *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, june 2011, pp. 381 –386.

[12] H. A. Baoxian Zhao and D. Zhu, "Enhanced reliability-aware power management through shared recovery technique," in *Computer-Aided Design - Digest of Technical Papers, 2009. ICCAD 2009. IEEE/ACM International Conference on*, nov. 2009, pp. 63 –70.

[13] T. Burd and R. Brodersen, "Energy efficient cmos microprocessor design," in *System Sciences, 1995. Proceedings of the Twenty-Eighth Hawaii International Conference on*, vol. 1, jan 1995, pp. 288 –297 vol.1.

[14] J. Ziegler, "Trends in electronic reliability: Effects of terrestrial cosmic rays," 2004. [Online]. Available: http://www.srim.org/SER/SERTrends.htm

[15] R. Xu, D. Mossé, and R. Melhem, "Minimizing expected energy consumption in real-time systems through dynamic voltage scaling," *ACM Trans. Comput. Syst.*, vol. 25, December 2007. [Online]. Available: http://doi.acm.org/10.1145/1314299.1314300