Exploration of Memory-Aware Dynamic Voltage Scheduling for Soft Real-Time Applications

Young-Jin Kim and Jihong Kim
School of Computer Science and Engineering
Seoul National University, Seoul Korea, 151-742
{youngjk, jihong}@davinci.snu.ac.kr

Abstract

Dynamic voltage scaling (DVS) and dynamic power management (DPM) are widely-used techniques to reduce energy consumption in modern computing systems. Although combining these techniques can save more energy, there has not been much work focused on energy-optimal combination of these techniques under variable memory clock frequencies. In this paper, we explore system-wide energy-optimal frequency space for a memory-aware DVS technique based on a stochastic memory access model for systems with frequency-variable memory devices. In addition, we propose a simple but practical DVS method, which can be applied to an actual platform.

1. Introduction

In modern computing systems, dynamic voltage scaling (DVS) and dynamic power management (DPM) are representative low-power techniques. As processor energy may take a high percentage of the total system energy consumption and memory chips are reported to be high power consumers in portable computing systems, several memory-aware DVS algorithms have been proposed for system-wide low-energy consumption [1, 2, 3]. For example, in [2], synergetic effects are investigated in combining DVS techniques and dynamic memory management policies.

Most of existing works, however, assume that external memories are based on a fixed supply voltage and a fixed clock frequency. With more recent embedded microprocessors (e.g., XScale), this assumption does not hold any more because these processors can adjust the memory clock frequency as well as the CPU clock frequency. In this paper, we address the problem of memory-aware DVS for systems with *variable memory clock frequencies*.

To the best of our knowledge, [4] is the only published work which addressed the combined DVS and DPM for the system with a variable memory clock frequency. However, the DPM scheme used in [4] considers only one type of slack interval, an idle interval after a task execution is completed. It also does not consider memory access patterns. Our work explores the energy-optimal frequency space using a DVS technique with a threshold-based DPM scheme depending on a stochastic model of memory accesses. In addition, we propose a simple but practical memory-aware DVS method. Experimental results using an MPEG-4 application shows the feasible energy-optimal frequencies vary depending on the degree of DPM energy saving. Our technique is shown to reduce the total energy consumption by more than 34% over the existing technique.

The rest of this paper is organized as follows. We describe a main motivation of our work in Section 2 while the energy model for memory-aware DVS and DPM is presented in Section 3. We explore the feasible energy-optimal frequency space for an MPEG-4 application in Section 4. Section 5 concludes with a summary.

2. Motivation

A feasible energy-optimal frequency assignment with a frequency-variable memory device indicates that a pair of a lower processor frequency and a lower memory frequency does not always result in a lower total energy consumption [4]. This is because the memory energy consumption can vary significantly depending on the selected processor and memory frequency pair, which affects the static and dynamic energy consumption of the memory device.

In Fig. 1, the left graph shows the original feasible solution space of the energy-optimal clock frequency setting on an MPEG-4 decoder [4]. The center of the contours shows a globally energy-optimal point. Each contour line gives the same total energy consumption. The deadline is always met in the upper region of the solid curve. In the left graph, point (a) is a frequency assignment ignoring both memory energy and access time. Point (b) considers only memory access times while point (c) takes into account both memory energy and access times. (Point (c) is the feasible energy-optimal point.)

Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'05)
1533-2306/05 $20.00 © 2005 **IEEE**

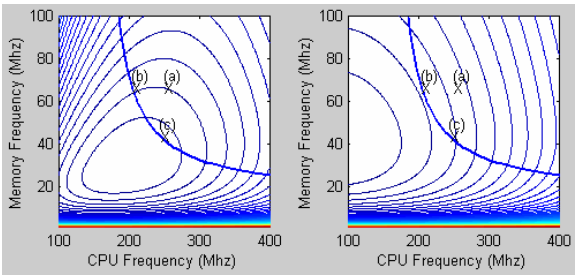

Figure 1. Original [4] and changed feasible solution space with a more aggressive DPM

In the meantime, the right graph in Fig. 1 shows the changed frequency space when the memory energy consumption is managed by a threshold-based DPM mechanism. In order to apply a deeper DPM mechanism, we augmented the original energy model [4] to enable a memory device to change into a lower power state after a specified threshold time at an idle state, regardless of task execution. We estimated the average interarrival time of memory accesses using the processor execution cycles, memory access count, and the deadline because memory traces were not available in [4]. In addition, we built a system-wide energy model based on the exponential distribution using the above average interarrival time and applied a threshold-based memory management policy.

We noticed that contours in the right graph were significantly changed over ones in the left graph, making point (b) nearer to the energy-optimal point than point (c). That is, although the frequency pairs are the same as those in the original frequency space, the energy consumption at (b) actually became lower than at (c) in the right graph. This phenomenon occurs because of two reasons. First, a more aggressive DPM reduces the static memory energy. Second, with a decreased static memory consumption, the total energy consumption becomes a function of the CPU frequency only. Hence, in order to develop an efficient memory-aware DVS policy, it is important to take account of the effect of the DPM policy adopted for the system.

3. Energy model for memory-aware DVS and DPM

3.1. Memory states and access model

We assume that external memory is an SDRAM device, usually having three power states, active, idle and powerdown [4]. We also assume external memory has an auto-precharge mode. This auto-precharge mode immediately sends the SDRAM into idle mode after every burst-mode access. The powerdown state is assumed not to perform any refresh operations. For the memory access model, we assume that the interarrival time between successive memory accesses follows an exponential distribution. (Although this may not exactly match with actual memory access behavior, it was known that using an exponential distribution produces almost the same result as one from using a simulation [3, 5].

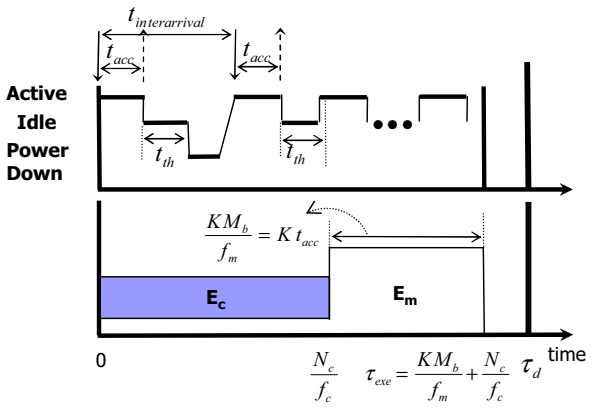

Figure 2. Processor and memory behavior

3.2. Energy model and energy-optimal solutions

We assume that a soft real-time task with the deadline τ_d executes N_c cycles at the clock frequency, f_c, and accesses the external memory K times. Figure 2 illustrates the processor and memory behavior we assumed for this study, where t_{acc} is the average delay in accessing the memory when each cache miss occurs. The memory switches from the idle state to the powerdown state after staying at the idle state for the specified threshold, t_{th}. $t_{interarrival}$ is the interarrival time between consecutive memory requests. Furthermore, the memory is assumed to operate with the memory clock frequency, f_m, and have a burst mode with the burst-mode transition cycles, M_b. τ_{exe} is the total execution time of the task.

Our processor energy model is the same as that in [4]. For the tractable memory energy model, we assume the total number of external memory accesses is given at a fixed value, K, and the distribution of memory accesses follows an exponential distribution with the access arrival rate, λ. (Due to the limited space, we omit the memory energy model. Refer to [8] for details.) To get optimal energy savings while the deadline is met, we should delay the total execution time until it reaches the deadline, that is,

$$\tau_{exe} = \frac{KM_b}{f_m} + \frac{N_c}{f_c} \le \tau_d \qquad (1)$$

With the given constraint (1), our goal is to find the optimal-energy frequency pair (f_c, f_m) such that the total energy consumption (given by Eq. (2)) is minimized. This formula can be considered as a generalized version of one described in [4], having more chances to power down to save more energy consumption. We can derive the energy-optimal frequencies by simultaneously solving equations $\partial E_t/\partial f_c = 0$ and $\partial E_t/\partial f_m = 0$.

$$
\begin{aligned}
E_t = & \left(\alpha C_c k^2 N_c + E_{IP} \frac{K\lambda^2}{2 f_{cmax}^2} (t_{acc} + t_{th})^2 \right) f_c^2 + E_{PI} \frac{K\lambda^2}{2 f_{cmax}^2} (t_{acc} + t_{th})^2 f_c^2 \\
& + \left(P_{PS} \frac{K\lambda}{2 f_{cmax}} (t_{acc} + t_{th})^2 - P_{IS} \frac{K\lambda}{2 f_{cmax}} (t_{acc} + t_{th})^2 \right) f_c \\
& - \left(E_{II} f_m \frac{K\lambda}{2 f_{cmax}} (t_{acc} + t_{th})^2 + E_{IP} \frac{K\lambda}{f_{cmax}} (t_{acc} + t_{th}) \right) f_c \\
& - E_{PI} \frac{K\lambda}{f_{cmax}} (t_{acc} + t_{th}) f_c + P_{PS} \left(\frac{K f_{cmax}}{\lambda} - N_c \right) \frac{1}{f_c} + E_{IA} K \\
& + E_{AI} K + E_{II} f_m K (t_{acc} + t_{th}) + P_{AS} \frac{K M_b}{f_m} + P_{IS} K (t_{acc} + t_{th}) \\
& - P_{PS} K (t_{acc} + t_{th}) + E_{IP} (K+1) + E_{PI} (K+1) + P_{PS} \tau_d - P_{PS} \frac{K M_b}{f_m}
\end{aligned} \tag{2}
$$

4. Exploring the feasible frequency space

In Section 3, we showed how to derive energy-optimal frequency equations using the energy model. An optimal solution can be obtained by solving high degree simultaneous nonlinear equations. However, since obtaining an exact solution can be complex and incur a large computational overhead, this method is not suitable for on-line voltage scheduling in low-power embedded systems. As an alternative, we explore the solution space extensively to understand the effects of system parameters on the energy-optimal frequency pair, and propose a practical memory-aware DVS method based on searching the feasible energy-optimal frequency space.

4.1. Exploring the energy-optimal space

We assume that the target system consists of a 32-bit RISC processor whose frequency can vary between [100Mhz, 400Mhz] and an SDRAM device with a maximum frequency of 100Mhz. We obtained memory traces with Simplescalar-ARM [6] for an MPEG-4 decoder selected as a target soft real-time application. Using the carphone.mp4 file, we profiled memory traces and built an exponential model based on the traces. Table 1 summarizes all the basic parameters of the SDRAM device model [4], the memory access model, and so on.

We searched the frequency space as the processor frequency and the memory frequency vary with regard to the threshold and other additional parameters [8]. Fig. 3 shows the transition of the energy-optimal point in the frequency space as the threshold t_{th} changes. The left upper graph shows the energy-optimal frequency space when the existing DVS method of [4] is applied. The right upper, the left lower, and the right lower graph show the frequency spaces of the proposed DVS method with 10 μs, 1 μs, and 300 ns as t_{th}, respectively. The left upper graph can be regarded as a special case of the proposed method with $t_{th} = \infty$. In other words, our energy model is more generic than the previous model and the powerdown state is never entered before the slack time occurs in the previous model. As t_{th} becomes smaller we notice that the total energy becomes a convex function of only the processor frequency. The solid lines in the graphs of Fig. 3 show the deadline constraint of Eq. (1). Therefore, we notice that the feasible energy-optimal frequency solution meeting the deadline is moved from about (235Mhz, 9Mhz) to about (221Mhz, 51Mhz) when we compare the left upper graph with the right lower one. This is caused by substantial reduction in the static memory energy consumption due to adjustment of the threshold which decides when the memory device should enter the powerdown state.

Table 1. Basic experimental parameters

E_{IA}	110.8 (nJ/access)	E_{AI}	15.8 (nJ/access)	M_b	9
E_{II}	3.14 (nJ/cycle)	E_{IP}	0	N_c	7.09×10^6 (cycles)
E_{PI}	0	P_{AS}	0.151 (W)	K	2,755
P_{IS}	0.072 (W)	P_{PS}	0.0116 (W)	$1/\lambda$	5488.23 (ns)
t_{acc}	60 (ns)	t_{th}	300 (ns)	$f_{m,\,max}$	100 (Mhz)

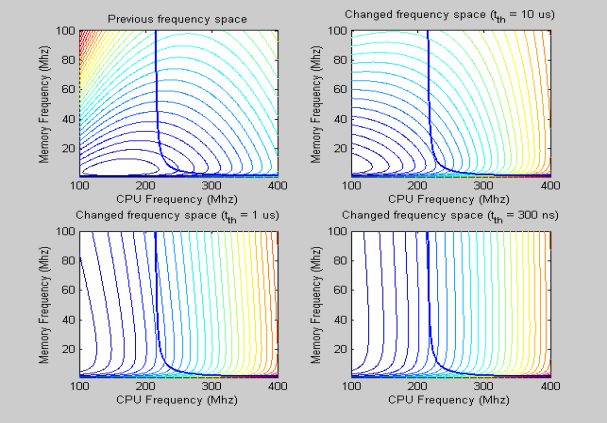

Figure 3. Variation of the energy-optimal frequency space with the threshold changed

4.2. Practical memory-aware DVS

We propose a simple but practical memory-aware DVS method by profiling the energy-optimal feasible frequency pairs off-line as memory access parameters vary. Our method is inspired by an event-driven, cruising DVS mechanism per process in [7].

First, we take three factors, $1/\lambda$, N_c, and K which govern system behaviors and model memory accesses, because we assume the threshold is already decided. (The threshold is fixed at 300 ns.) We build a table for the parameters within proper ranges by specified step sizes, while the deadline and the bounds of frequencies are met. $1/\lambda$ varies from 300 ns to 6,000 ns with 20 steps. N_c and K vary from 4×10^5 to 8×10^6 with 20 steps and from 2,000 to 42,000 with 5 steps, respectively. Then, we obtain an energy-optimal frequency pair of the next frame using the

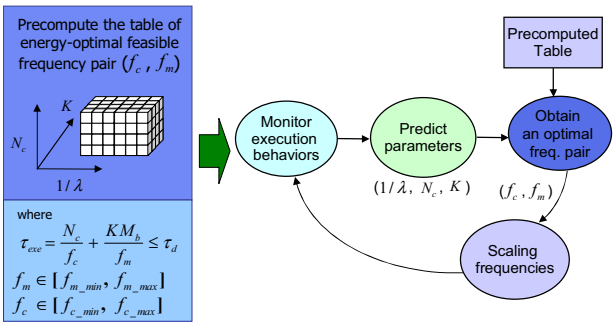

Figure 4. Procedure of a practical memory-ware DVS

Table 2. Energy-optimal frequencies and energy consumption

DVS method	f_c (Mhz)	f_m (Mhz)	E_c (μJ)	E_m (μJ)	E_t (μJ)	Energy saving (%)
Non-DVS	400	100	11,344	7,400	18,744	0
Previous [4]	221	51	3,463	7,872	11,335	38.4
	235	9	3,916	3,788	7,704	58.9
Proposed	221	51	3,463	957	4,420	76.4
	235	9	3,916	1,148	5,064	72.9

pre-computed table indexed by $1/\lambda$, N_c, and K and scale the processor frequency and the memory frequency accordingly. The overall procedure of our heuristic method is shown in Fig. 4.

In order to evaluate the performance of the proposed DVS heuristic, we assumed that each frame satisfies the following ranges for three prediction parameters: $1/\lambda \in$ [5400ns, 5700ns], $N_c \in$ [2000, 12000], and $K \in [6.8\times10^5, 7.2\times10^6]$. Once three parameters are estimated for the next frame, the precomputed table was searched using the estimated parameters. The final frequency pair was computed by interpolation of nearest neighboring table entries.

In Table 2, the frequency pair of (235Mhz, 9Mhz) is the feasible energy-optimal solution of the previous method in [4]. (221Mhz, 51Mhz) represents the feasible frequency pair of the proposed method. We notice that the optimal energy frequency pair is moved to around f_c = 221Mhz and f_m = 51Mhz. We also notice that energy saving rates are higher than those of the previous method by at least 14%.

5. Conclusions

For modern embedded microprocessors with variable memory clock frequencies as well as variable CPU clock frequencies, a memory-aware DVS technique is necessary. In order to develop such a system-wide low-power technique, we explored the energy-optimal frequency pair space using an MPEG-4 application.

The main result of our analytical study is that the optimal frequency pair for a given system can vary significantly depending on the degree of DPM energy saving. In particular, we showed the optimal frequency pair of the existing technique changes to a more close-to-optimal one using our approach as the powerdown threshold changes. We also proposed a simple but practical memory-aware DVS heuristic which saves more than 34% energy consumption over the existing method.

Acknowledgement

This research was supported by the Ubiquitous Autonomic Computing and Network Project, 21st Century Frontier R&D Program in Korea and was also supported by University IT Research Center Project.

References

[1] T. L. Martin, and D. P. Siewiorek, "Nonideal battery and main memory effects on CPU speed on CPU speed-setting for low power", *IEEE Transactions on VLSI Systems*, vol. 9, no. 1, pp. 29–34, Feb 2001.

[2] X. Fan, C. S. Ellis, and A. R. Lebeck, "The synergy between power-aware memory systems and processor voltage", in *Proc. of Power-Aware Computer Systems*, 2003.

[3] T. Simunic, L. Benini, A. Acquaviva, P. Glynn, and G. De Micheli, "Dynamic Voltage Scaling and Power Management for Portable Systems", in *Proc. of Design Automation Conference*, 2001, pp. 524-529.

[4] Y. Cho and N. Chang, "Memory-Aware Energy-Optimal Frequency Assignment for dynamic Supply Voltage Scaling", in *Proc. of International Symposium On Low Power Electronics and Design*, 2004, pp. 387–392.

[5] X. Fan, C. S. Ellis, and A. R. Lebeck, "Memory Controller Policies for DRAM Power Management", in *Proc. of International Symposium on Low Power Electronics and Design*, Aug. 2001, pp. 129-134.

[6] D. Burger and T.M. Austin, "The Simplescalar Tool Set, Version 2.0", *Computer Architecture News*, pages 13–25, June 1997.

[7] A. Weissel and F. Bellosa, "Process cruise control: event-driven clock scaling for dynamic power management", in *Proc. of International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, Oct. 2002, pp.238-246.

[8] Y.-J. Kim and J. Kim, "Exploration of Memory-Aware Dynamic Voltage Scheduling for Soft Real-Time Applications", Technical Report, SNU, May 2005.
Available at *http://cares.snu.ac.kr/Download/rtcsa05_tr.pdf.*