Energy-Aware Scheduling for Tasks with Mixed Energy Requirements

Mario Bambagini, Giorgio Buttazzo
{mario.bambagini, giorgio.buttazzo}@sssup.it
Scuola Superiore Sant'Anna, Pisa, Italy

Marko Bertogna
marko.bertogna@unimore.it
University of Modena and Reggio Emilia, Italy

Abstract

Two widespread techniques, Dynamic Voltage/Frequency Scaling (DVFS) and Dynamic Power Management (DPM), rely on speed reduction and low-power state exploitation, respectively, to reduce energy consumption in modern computing platforms. Traditionally, dynamic power was the main source of processor power consumption, therefore DVFS was more effective than DPM to reduce energy consumption in processing elements, whereas DPM was mainly used to shut down system components not currently in use. As the technology continues to miniaturize processors, leakage power has become dominant with respect to dynamic power consumption, hence today most people believe that DPM is the most effective technique also in processing units.

This paper shows that, considering a more realistic task model derived from the analysis of actual executed code, tasks may exhibit different energy requirements, so requiring an integrated DVFS-DPM approach.

I. Introduction

Two widely used techniques to save energy are *Dynamic Voltage/Frequency Scaling* (DVFS) and *Dynamic Power Management* (DPM). DVFS approaches [1] decrease the voltage and/or frequency of the processor to reduce energy consumption, while DPM techniques [2] aim at putting the processor in a low-power inactive state as long as possible, but still guaranteeing tasks timing constraints. Nowadays, DPM techniques are more effective for reducing the overall energy consumption as they significantly affect the leakage dissipation whose impact has become dominant in actual hardware due to shrunk transistor size and low supply voltage. However, such energy reduction is obtained by taking into account only hardware features, without considering any characteristic related to the application. For instance, for intensive I/O-bound applications, task execution times are less dependent on the processor speed, hence the energy consumption ascribable to the leakage dissipation is barely affected by speed changes, offering the possibility of achieving higher reductions due to dynamic consumption by scaling the speed.

Experimental measurements show that DPM techniques work better for CPU bound tasks, while DVFS techniques are more appropriate for I/O bound tasks. Hence, this paper suggests to enrich the task model to consider the types of operations carried out by tasks, to apply the most appropriate technique or integrate them to achieve a better performance.

This paper also introduces several ideas for dealing with applications with mixed energy requirements. To the best of our knowledge, the state of art algorithms that cope with tasks and devices together consider only DVFS features for the task set and low-power states for the devices [3]. Moreover, tasks with different energy requirements was analyzed by Aydin et al. [4], but the solution considered only a DVFS approach for CPU bound tasks with different critical speeds.

II. System model

We consider a set Γ of n sporadic tasks $\tau_1, \tau_2, \ldots, \tau_n$ executing upon a single processor. The processor can vary the running speed s, defined as the normalized frequency with respect to the maximum frequency, $s = \frac{f}{f_{max}}$. The speed set is assumed to be finite and composed of m different speeds $s_1, s_2, \ldots, s_m$ sorted in ascending order, thus $s_{min} = s_1$ and $s_{max} = s_m = 1.0$.

Each sporadic task τ_i ($1 \leq i \leq n$) is characterized by a worst-case execution time (WCET) $C_i(s)$, which is a monotonic not decreasing function of the speed, a relative deadline D_i and a minimum inter-arrival time T_i, also referred to as the period. The WCET value of τ_i depends on the actual speed of the processor and is computed as $C_i(s) = \alpha_i C_i^{max} + (1 - \alpha_i)\frac{C_i^{max}}{s}$, where C_i^{max} denotes the amount of time required to execute τ_i at the maximum speed and $\alpha_i \in [0, 1]$ represents the fraction of execution time that does not scale with the speed (e.g., due to I/O operations). The larger α_i, the bigger the portion of τ_i that does not scale with the speed.

The power consumption of each gate depends on the supply voltage V and clock frequency f as reported in Equation 1:

$$P_{gate} = C_L V^2 p_s f + p_s V I_{short} + V I_{leak}. \quad (1)$$

A generic formulation to model the power consumption of the entire processor in the active state has been proposed by Martin et al. [5] as a function of the running speed, $P(s) = K_3 s^3 + K_2 s^2 + K_1 s + K_0$.

Moreover, the processor provides a set of low-power states during which the task execution is suspended while the power consumption is low. Typically, sleep states characterized by a lower power consumption have longer wake-up times to restore the fully operating state. This feature leads to discarding several low-power states when the available idle time is shorter than such a wake-up time (also referred to as break-even time).

III. Motivational Example

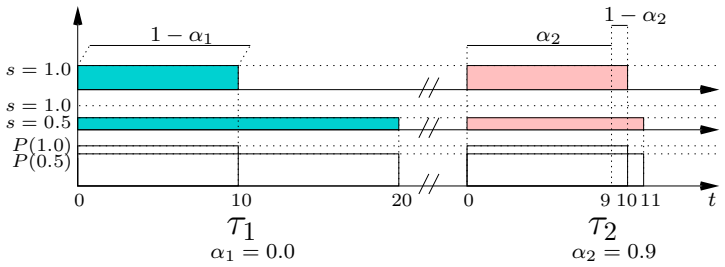

Fig. 1. Execution and consumption of τ_1 and τ_2 at speed 1.0 and 0.5

This section provides an example that shows how the type of code executed by a task (either CPU or I/O bound) can affect the strategy to minimize energy consumption, between pure DPM, according to which the task is executed at the maximum speed to exploit low-power states, and pure DVFS, where the processor is set at a slower speed reducing idle intervals.

Let us consider two tasks τ_1 and τ_2 with computation times $C_1(1.0) = C_2(1.0) = 10$, $\alpha_1 = 0$ and $\alpha_2 = 0.9$.

Concerning the hardware, let us consider a NXP LPC1768 equipped with an ARM Cortex M3. Such a processor supports frequencies within $[36, 96]$ MHz with steps of 4 MHz and the normalized power function is $P(s) = 0.4s + 0.6$ ($s_m = 1.0$ corresponds to 96 MHz). Note that, since this processor does not support voltage scaling, according to Equation 1 the power consumption can be considered a linear function of the speed. However, a quadratic or cubic power function would only decrease the impact of the dynamic power component. Considering the normalized power function reported above, 60% of the overall power consumption is due to static dissipation, while the remaining 40% is affected by the speed dependent part (which may be linear, quadratic or cubic).

For this architecture, the energy required for the execution of CPU bound code is minimized when the system runs at the maximum speed. For the sake of simplicity, let us exploit only the frequencies 48 and 96 MHz ($s_1 = 0.5$ and $s_2 = 1.0$).

The two values for α are obtained by executing, on the platform in use, the Coremark benchmark ($\alpha = 0.0$) and a test program that heavily exchanges data within an external storage unit ($\alpha = 0.9$).

Figure 1 shows the execution of the two tasks for different processor speeds with the corresponding power consumption. Note that when τ_1 (characterized by $\alpha_1 = 0.0$) executes at speed $s = 1.0$, its computation time is $C_1(1.0) = 10$ and the corresponding energy consumed by the processor is $10 \times P(1.0) = 10$. When the same task executes at speed $s = 0.5$, its computation time is $C_1(0.5) = 20$ and the consumed energy is $20 \times P(0.5) = 16$. For task τ_2 (with $\alpha_2 = 0.9$) the situation is quite different. Indeed, at speed $s = 1.0$, we have $C_2(1.0) = 10$, consuming an energy equal to $10 \times P(1.0) = 10$, whereas at speed $s = 0.5$, we have $C_2(0.5) = 11$, consuming an energy equal to $11 \times P(0.5) = 8.8$.

Hence, for CPU bound tasks, like τ_1, the energy is minimized at the highest speed $s = 1.0$, whereas for I/O bounded tasks, like τ_2, energy is minimized at lower speed ($s = 0.5$ in the example), leaving space for DVFS techniques.

IV. Open Questions

Given the different energy characteristics of CPU and I/O bound tasks, the open problem to be investigated is then to find energy-efficient scheduling strategies that mix DPM and DVFS approaches for exploiting the characteristics, in terms of computational time and energy-efficiency, of the software that composes the task set.

Two possible approaches have been identified, which may lead to a further energy saving.

The first approach consists of extrapolating from the task set and platform a generalized parameter representing the dominant behavior to find out which kind of technique is more appropriate (either DPM or DVFS). More precisely, such a parameter would be useful to select at design-time the most appropriate technique (DPM or DVFS).

A second and more sophisticated approach may exploit speed scaling according to the task in execution [4] (for instance, the speed would increase at the maximum value when running a CPU bound task). In practice, scaling speed for every task introduces a significant overhead [6] and affects the system reliability [7]. For overcoming these drawbacks the algorithm could try to compact the execution of tasks which belong to the same category, so reducing the number of speed scaling events.

References

[1] H. Aydin, P. Mejía-Alvarez, D. Mossé, and R. Melhem, "Dynamic and aggressive scheduling techniques for power-aware real-time systems," in *Proceedings of the 22nd IEEE Real-Time Systems Symposium*, 2001.

[2] M. A. Awan and S. M. Petters, "Enhanced race-to-halt: A leakage-aware energy management approach for dynamic priority systems," in *Proceedings of the 2011 23rd ECRTS*, 2011.

[3] V. Devadas and H. Aydin, "On the interplay of dynamic voltage scaling and dynamic power management in real-time embedded applications," in *Proceedings of the 8th ACM international conference EMSOFT*, 2008.

[4] H. Aydin, V. Devadas, and D. Zhu, "System-level energy management for periodic real-time tasks," in *RTSS*, 2006.

[5] T. Martin and D. Siewiorek, "Non-ideal battery and main memory effects on cpu speed-setting for low power," *IEEE Transactions on VLSI Systems*.

[6] M. Bambagini, F. Prosperi, M. Marinoni, and G. Buttazzo, "Energy management for tiny real-time kernels," in *ICEAC*, 2011.

[7] B. Zhao, H. Aydin, and D. Zhu, "Energy management under general task-level reliability constraints," in *RTAS*, 2012.