Unified, Ultra Compact, Quadratic Power Proxies for Multi-Core Processors

Muhammad Yasin, Anas Shahrour, Ibrahim (Abe) M. Elfadel
Institute Center for Microsystems (iMicro)
Masdar Institute of Science and Technology
Abu Dhabi, UAE

Abstract—**Per-core power proxies for multi-core processors are known to use several dozens of hardware activity monitors to achieve a 2% accuracy on core power estimation. These activity monitors are typically not accessible to the user, and even if they were accessible, there would be a significant overhead in using them at the kernel or OS level for power monitoring or control. Furthermore, when scaled up to hundreds of cores per chip, such power proxies become a computational bottleneck for power management operations such as chip power capping. In this paper, we show that a 4% accuracy or better for per-core power estimation can be achieved using an ultra compact power proxy based on a hybrid set of only four user-accessible parameters, namely core frequency, core temperature, instruction-per-cycle and active-state residency. Our proxy is nonlinear, valid across all P and C states, and is based on a randomized power data collection strategy that aims at exercising all the P and C levels of each core. We illustrate the accuracy of the model using the full suite of the SPEC CPU 2006 benchmarks on a 12-core processor.**

I. Introduction

An intuitive and logical way to implement power management and control policies in multicore systems is at the granularity of a core. Unfortunately, direct measurement of per-core power using hardware power meters is not scalable for large number of cores [1] as it creates demand for sampling/ aggregating large amounts of data and coordination among multiple power metering devices. Even in cases where some support for per-core power metering exists, no such information is accessible at the OS or application level. A case in point is the power proxy used in IBM's eight-core Power7 processor [2] where more than 40 hardware activity monitors per core are used as input to a core power proxy that is linear in the values of the monitors and within 2% accuracy. These activity monitors are not user-accessible. Our work is focused on developing and validating ultra compact, per-core power proxies for multi-core processors using a very small set of PCM's that are all user-accessible. Our ultimate goal is to use such models for developing scalable, control-theoretic policies [3], [4] for power management of mega core processors

In a recent paper [5], we used Intel's Running Average Power Limit (RAPL) interface to develop linear and quadratic models for per-core dynamic power consumption based on performance monitoring counters. We validated the models on an industrial platform, namely Dell's PowerEdge T620 server equipped with two 6-core Intel Xeon E5-2630 processors. The power models we developed are of two different categories: a linear model for the case when sleep states (also called C-states) are disabled and a quadratic model for the case when sleep states are enabled. We have found that a linear model with frequency and IPC alone can estimate power to high accuracy when the sleep states are disabled. However, when the cores are allowed to enter power-saving sleep states, we have used active-state residency in the power proxy as a modulator on other PCM's. This has resulted in a novel quadratic power proxy in the PCM's, which is accurate to within 4% and having an order of magnitude less parameters than a state-of-the-art per-core power proxy [2].

In this follow-up paper, we combine the linear and quadratic models to develop a single unified model for per-core power consumption representing both usage scenarios for the sleep states. The unified model is nonlinear, and the C0RES parameter (active-state residency) is used to modulate all other PMC values. The unified nonlinear model uses only four PMC's and is accurate to within 3.30%. A distinguishing feature of our model is its ability to accurately predict per-core power across all P (performance) and C (sleep) states. This is to be contrasted with [1] where a separate power model is needed for each DVFS level and where sleep states were not considered. In order to broaden the validity domain across all P and C states, we have used a novel randomized data collection strategy that has insured the exercising of all DVFS and sleep levels for each core in the processor.

The rest of this paper is organized as follows. Section II reviews power modeling using performance monitoring counters. Section III describes the modeling and evaluation methodology, especially as it relates to the sleep states of each core. In section IV, we describe our novel data collection strategy and present the experiments performed as well as the validation results. Finally, conclusions and future work are given in section V.

II. Related Work

Bellosa *et al.* [6] presented the idea of treating power as a resource and used hardware performance monitoring counters to estimate power consumption at particular frequencies. Bircher *et al.* [7] identified the correlations of certain PMCs with power and determined under what conditions PMCs like IPC could effectively represent power consumption. In the context of per-core modeling in multicore systems, Goel *et*

978-3-9815370-2-4/DATE14/©2014 EDAA

al.[1] have developed piecewise linear per-core power models for systems with up to 8 cores. A correlation-based scheme is used for selecting appropriate performance counters. The resulting power model is specific to a particular frequency. Takouna *et al.* [8] have found a strong linear relationship between the number of active cores and power consumption. Their power consumption model for a multi-core processor uses the average operating frequency and the number of active cores. None of the reported power models accounts for the effect of sleep states on power consumption.

III. Methodology

This section describes PMCs and power monitoring, the criteria for PMC selection, and the evaluation methodology.

A. Monitoring Power and PMCs

All modern processors, especially the mobile platforms come with some sort of power management support at levels varying from hardware to OS-level. For example, Intel Sandy Bridge processors come with digital energy metering and power limiting capabilities through the Running Average Power Limit (RAPL) interface [9]. The focus of our work is to utilize the per-socket energy consumption provided by the RAPL interface and estimate power consumption on a per-core basis.

In order to access RAPL energy values, we make use of the Intel Performance Counter Monitor (PCM) [10]. Intel PCM is a tool that supports monitoring of hardware performance counters on Intel processors. The per-core performance counters monitored include: EXEC, IPC, AFREQ, L2MISS, L3MISS, L2HIT, L3HIT, residencies for C-states, and TEMP. EXEC corresponds to instructions per nominal CPU cycle whereas IPC refers to Instructions per CPU cycles where only the cycles during which CPU is active are counted. TEMP represents core temeprature.

B. Effect of Sleep States

Modern processors support ACPI-compliant sleep states. Depending upon the perceived duration of idleness, a processor may turn off its components resulting in significant power savings. Sleep states are also referred to as C-states. State C0 is the active or un-halted state. Deeper sleep states are indicated by higher C indices. The higher the index the more processor components are turned off in the corresponding sleep state. In a multicore processor, some cores can be put to partial or complete sleep based on the workload.

C. Per-core Power Model

Let us denote by P_N the total power of N active cores. Then we have:

$$P_N = \sum_{n=1}^{N} P_c(n) \; with \;\; P_c(n) = \sum_{i=1}^{K_n} \alpha_{ni} C_{ni} \qquad (1)$$

where $P_c(n)$ represents the power consumed in the n-th core and is assumed to be a linear combination of K_n performance counters, $C_{ni}, 1 \leq i \leq K_n$, and the α_{ni} are the weighting coefficients.

D. Evaluation Methodology

We develop and validate per-core power models across 30 programs from the SPEC CPU2006 benchmark suite. The benchmark suite comprises both integer and floating point programs. SPEC CPU2006 benchmarks are inherently single-threaded. We execute multiple copies of a benchmark program in parallel to make multiple core execute some tasks. We use least-squares regression to find the model's coefficients, and 4-fold cross validation scheme for model validation. The data collected is repeatedly divided into training and test datasets. In each iteration, the model coefficients are computed based on training data and validated on the test data using mean absolute error (MAE). The final model coefficients and modeling error are the average of values over the 4 iterations.

IV. Experiments and Results

A. Platform Specifications

We conducted all of our experiments on Dell PowerEdgeT620 server running Red Hat Enterprise Linux Server 6.1 with Linux kernel 2.6.32. The server has two Intel Xeon E5-2630 processors, each with 6 cores, for a total of 12 cores.

Intel® PCM gives us two power measurements, one for each socket or processor. Data is collected from PCM in csv format and Matlab is used for statistical analysis and regression. Xeon E5-2630 processors support 12 discrete DVFS set points or P-states raning from 1.2GHz to 2.3GHz. To set the frequency of a core, we use the *CPUfreq* kernel infrastructure and userspace governor.

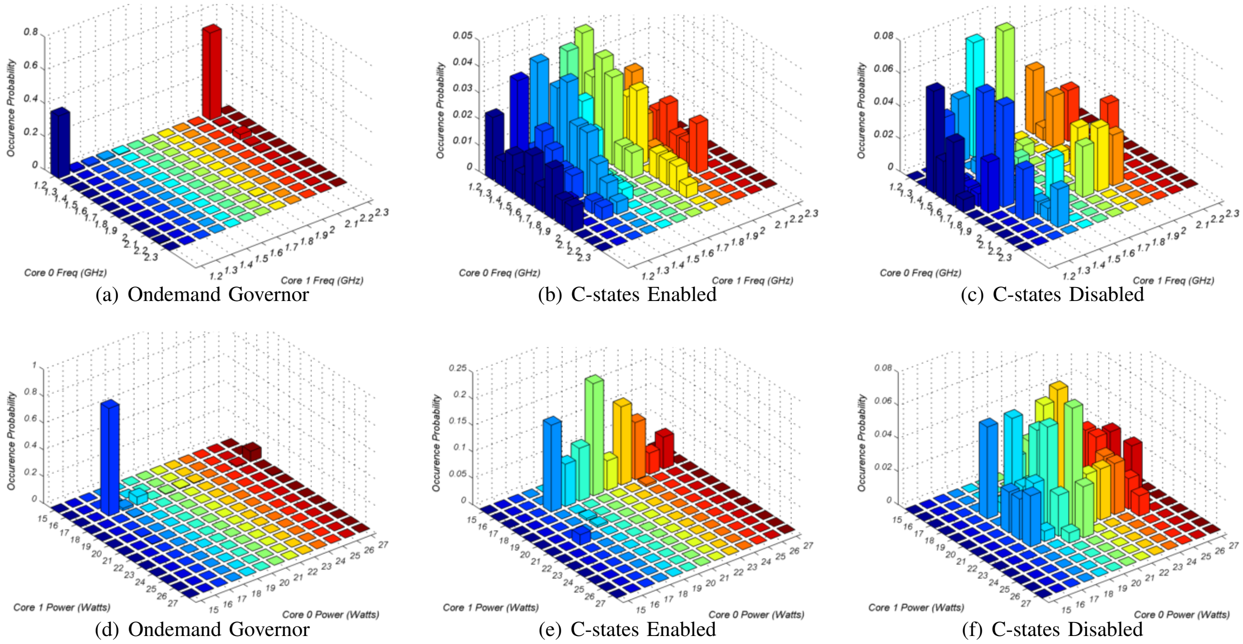
B. Data Collection

In contrast with [1] where a separate model is needed for each available DVFS, one of our main objectives is to develop a power model that is valid across all DVFS levels. This requires that we collect data at all possible combinations of core DVFS settings. Note that this is impossible to achieve using the default Linux kernel where conservative, performance or on-demand policies are pre-defined and used. It is interesting to note that the default behavior of the kernel, for all SPEC CPU2006 benchmarks, is to assign a corner DVFS to the core. As illustrated by the histogram in Figure 1(a), even for the on-demand governor, the least aggressive adaptive policy, often the cores operates at one of two extreme DVSF levels. Intermediate frequencies are rarely assigned as indicated by very low probability of occurrence.

For near-uniform coverage of the DVFS space, we assign DVFS levels to the cores randomly after regular intervals of five seconds throughout the execution of a benchmark. It is clear from the histograms in Figure 1(b) and 1(c) that this random DVFS assignment scheme can cover the entire DVSF space much more uniformly than can be done using the existing linux governors.

C. Impact of Sleep States

In case of multiple cores, not all cores have to be active at the same time. Depending on the characteristics of the workload under execution, some cores can be idle and thus enter



(a) Ondemand Governor (b) C-states Enabled (c) C-states Disabled

(d) Ondemand Governor (e) C-states Enabled (f) C-states Disabled

Fig. 1. Effective frequency (top) and power (bottom) histograms for the xalancbmk benchmark

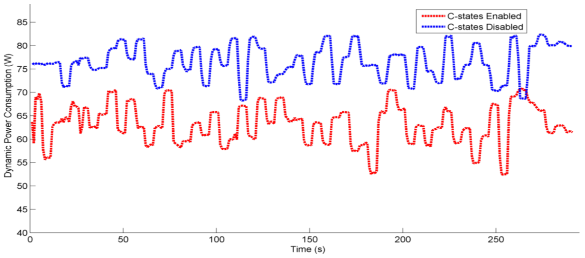


Fig. 2. Difference in power consumption with and without C-states

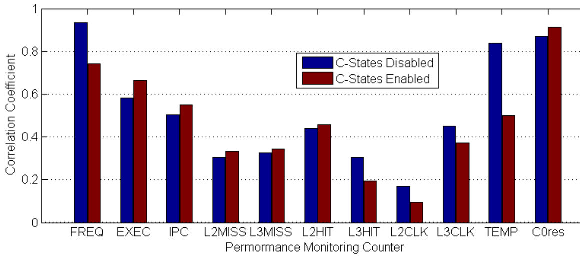


Fig. 3. Correlation of various PMCs with Processor Power

a sleep state. This directly translates into savings in power consumption. Figure 2 shows the total power consumption when all 12 cores are active and running the same workload. This is done for the two cases of sleep states enabled and disabled. The power reduction resulting from enabling C-states is clearly shown in Figure 2 and is the main motivation for developing a power proxy adapted to each C-state setting.

This reduction in total power consumption may be further explained by comparing frequency histograms in Figures 1(b) and 1(c) against the power histograms in Figures 1(e) and 1(f). For these experiments, two cores were active, one in each socket, and only one SPEC CPU2006 benchmark was running on the processor. When the C-states are disabled, all the cores are active and a change in frequency instantly leads to a change in power consumption. When C-states are enabled, however, one core can go idle, as can be observed in Figure 1(e) in terms of almost constant power consumption for core 1.

D. Performance Monitoring Counter Selection

The lower the number of performance counters selected for power proxy, the more scalable is model wit increasing number of cores. A very small number of counters, however, may result in a significant loss of modeling accuracy. We have therefore to identify the key performance counters that correlate best with power consumption.

Figure 3 shows the correlation between power consumption and the per-core PMCs. Power has high correlation with C0RES, EXEC, TEMP, IPC and operating frequency FREQ. These high correlation values are very intuitive. Enabling sleep states results in a lower correlation of power with frequency and temperature.

Including two PMCs which are not independent in the regression models may deteriorate the model accuracy, and in a linear model may typically results in a negative weight ($\alpha' s$) for one of the PMC's. Ranking based on correlation, we have chosen the following four performance counter for multicore power modeling: IPC, FREQ, TEMP and C0RES as they exhibit the highest correlation with core power consumption. The introduction of a temperature term will help account for leakage power and is in line with other models accounting for both dynamic and leakage power [3].

TABLE I: Coefficients for the unified model

Parameter	Value
α_{FREQ}	6.39
α_{IPC}	-0.16
α_{C0RES}	-0.08
α_{TEMP}	.02
α_0	41.34
MAE	3.30%

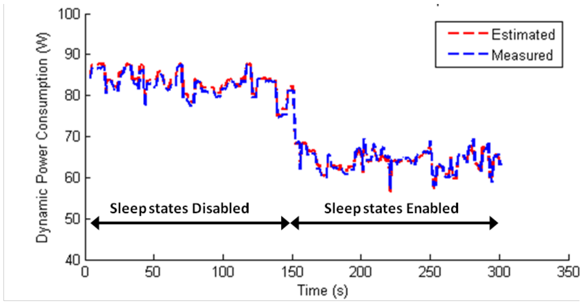


Fig. 4. Multicore power estimation with the unified power model for 12 active cores

E. Unified Power Models with 4 Performance Counters

In our previous work [5], we reported on two different power models, one for sleep states enabled and other for sleep states disabled. In this work, we present a single unified model which captures power consumption for both scenarios. In order to incorporate the effect of sleep states, we modulate the performance counters by the C-state residency parameter, namely, C0RES, as represented in

$$\begin{aligned} P &= \alpha_0 + \alpha_{TEMP} \sum_{n=1}^{N} C^{n}_{TEMP} C^{n}_{C0RES} \\ &+ \alpha_{FREQ} \sum_{n=1}^{N} C^{n}_{FREQ} C^{n}_{C0RES} \qquad (2) \\ &+ \alpha_{IPC} \sum_{n=1}^{N} C^{n}_{IPC} C^{n}_{C0RES} + \alpha_{C0RES} \sum_{n=1}^{N} C^{n}_{C0RES} \end{aligned}$$

where n indicates the core number, N is the total number of active cores, and P is the total power consumption of the processor. The power of the n^{th} core is the weighted sum of the performance counter values corresponding to the core. Modulating performance counter values by the C0RES parameter results in a nonlinear model. To compute its coefficients, nonlinear regression is used with optimal parameters given in Table I. It is worth nothing that when C-states are disabled, a core can not enter any of the sleep states, and the C0RES is 100%. The model becomes effectively linear. Figure 4 illustrates the accuracy of the model in tracking processor power through all sleep-state scenarios. The Mean Absolute modeling Error (MAE) for this unified model is only 3.30%.

V. Conclusion and Future Work

In this paper, we have developed a unified scalable per-core dynamic power proxy for multicore processor using Intel RAPL Interface. The main novelty of our power proxy is the introduction of sleep-state residency as an independent performance counter *and* as a modulator for other counters. The latter feature makes the model quadratic and results in improved accuracy especially when core sleep states are enabled. The model has been validated using SPEC CPU2006 benchmarks on a 12-core processor with MAE error of 3.3%. Default Linux kernel drivers and policies have been used for C-state transitions. Deeper insights can be obtained into power consumption behavior of C-states by customizing the Linux drivers. Only the performance counters supported by Intel PCM have been considered. In the future, we plan to use tools such as *pfmon* and integrate them with the Intel RAPL interface to extend the PCMs used. We also plan to train and test the new power proxy model using media, productivity and web-centric workloads.

Acknowledgment

The authors would like to acknowledge very helpful discussions with Andrew Henroid from Intel, and with Pradip Bose, Alper Buyuktosunoglu, and Canturk Isci from IBM. This work was supported by SRC under Contract 2011-TJ-2192 with customized funding from ATIC, Abu Dhabi, UAE.

References

[1] B. Goel, S. A. McKee, R. Gioiosa, K. Singh, M. Bhadauria, and M. Cesati, "Portable, scalable, per-core power estimation for intelligent resource management," in *Int. Green Computing Conference*, 2010.

[2] M. Floyd, M. Ware, K. Rajamani, T. Gloekler, B. Brock, P. Bose, A. Buyuktosunoglu, J. Rubio, B. Schubert, B. Spruth, J. Tierno, and L. Pesantez, "Adaptive energy-management features of the ibm power7 chip," *IBM Journal of Research and Development*, vol. 55, no. 3, pp. 8:1–8:18, 2011.

[3] W. Huang, C. Lefurgy, W. Kuk, A. Buyuktosunoglu, M. Floyd, K. Rajamani, M. Allen-Ware, and B. Brock, "Accurate fine-grained processor power proxies," in *Proceedings of the 45th Annual International Symposium on Microarchitecture*, December 2012, pp. 224–234.

[4] A. Bartolini, "Dynamic power management: from portable devices to high performance computing," Ph.D. dissertation, University of Bologna, 2011.

[5] M. Yasin, A. Shahrour, and I. Elfadel, "Ultra compact, quadratic power proxies for multi-core processors," in 20^{th} *IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, 2013.

[6] F. Bellosa, "The benefits of event: driven energy accounting in power-sensitive systems," in *Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system.* ACM, 2000, pp. 37–42.

[7] W. Bircher, J. Law, M. Valluri, and L. K. John, "Effective use of performance monitoring counters for run-time prediction of power," *University of Texas at Austin Technical Report TR-041104-01*, 2004.

[8] I. Takouna, W. Dawoud, and C. Meinel, "Accurate mutlicore processor power models for power-aware resource management," in *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on.* IEEE, 2011, pp. 419–426.

[9] A. Naveh, D. Rajwan, A. Ananthakrishnan, and E. Weissmann, "Power management architecture of the 2nd generation intel® core microarchitecture, formerly codenamed sandy bridge," 2011.

[10] T. Willhalm. (2012, August) Intel performance counter monitor - a better way to measure cpu utilization. [Online]. Available: http://software.intel.com/en-us/articles/intel-performance-counter-monitor-a-better-way-to-measure-cpu-utilization