Analysis of RAPL Energy Prediction Accuracy in a Matrix Multiplication Application on Shared Memory

✉Juan Manuel Paniego[1], Silvana Gallo[1,3], Martin Pi Puig[1], Franco Chichizola[1], Laura De Giusti[1], Javier Balladini[2]

[1] Instituto de Investigación en Informática LIDI (III-LIDI), Facultad de Informática, Universidad Nacional de La Plata, La Plata, Argentina
[2]Departamento de Ingeniería de Computadoras, Facultad de Informática, Universidad Nacional del Comahue, Neuquén, Argentina
[3]CONICET, Facultad de Informática, Universidad Nacional de La Plata, La Plata, Argentina

{jmpaniego, sgallo, mpipuig, francoch, ldgiusti}@lidi.info.unlp.edu.ar
javier.balladini@fi.uncoma.edu.ar

Abstract. In recent years, energy consumption has emerged as one of the biggest issues in the development of HPC applications. The traditional approach of parallel and distributed computing has changed its perspective from looking for greater computational efficiency to an approach that balances performance with energy consumption. As a consequence, different metrics and measurement mechanisms have been implemented to achieve this balance. The objective of this article focuses on monitoring and analyzing energy consumption for a given application through physical measurements and a software interface based on hardware counters. A comparison of the energy values gathered by Intel RAPL versus physical measurements obtained through the processor power source is presented. These measurements are applied during the execution of a classic matrix multiplication application. Our results show that, for the application being considered, the average power required by the processor has an error of up to 22% versus the values predicted by RAPL.

Keywords: Energy consumption, Prediction, Power, Hardware counters, RAPL, Perf**.**

1 Introduction

The goal of High Performance Computing (HPC) is to design and develop solutions to situations that require large computing capacities by using techniques and methods that allow using all available computing power to achieve maximum speedup and efficiency.

In recent years, due to existing limitations in energy resources, greater emphasis has been placed on researches related to the consequences of increasing processor temperature. The trend has been analyzing energy consumption on different applications and developing consumption models to predict and plan for execution based on energy consumption metrics [2, 3].

Currently, power consumption efficiency is a very important metric that affects everything from mobile systems to large HPC centers. Daily energy consumption in the latter is very high and this metric increase over time. Therefore, monitoring power dissipation and energy consumption in every system has become a priority research area.

Many biggest clusters currently operating are formed by general-purpose machines that, above all, have a low monetary cost. By interconnecting dozens, hundreds or thousands of these devices, significant computational power is obtained, albeit with very high power consumption, which includes the energy required for cluster nodes to operate as well as the energy required to keep such nodes refrigerated, for example. For this reason, monitoring the energy used for each of the computers is a key to be able to reduce this variable.

This article focuses on monitoring and analyzing energy consumption for a matrix multiplication application through physical measurements and a software interface based on hardware counters.

The article is organized as follows: Section 2 presents an overview of related works in the energy consumption area. In Section 3, some basic concepts of energy consumption measurement are defined. After that, Section 4 details the methodology used to carry out the measurements. Then, in Section 5, the experimental work is described, whereas in Section 6, the results obtained are presented. Finally, Section 7 presents our conclusions and future work.

2 Related work

Energy consumption in HPC has emerged as one of the major research areas in recent years due to the need of minimizing the environmental impact of the infrastructure required by computation centers, large servers, storage systems and refrigeration systems. To manage energy consumption, numerous applications and measurement instruments such as the ones used in this article have been designed.

When looking to determine a reference value corresponding to the consumption of an entire computation system, which is known as *coarse-grained measurements*, different instruments that intercept power sources and applications that extract the corresponding data are used. For example, external measurements with clamp probes and oscilloscopes [2, 3], or devices such as Watt's Up Pro combined with applications that take captured data and output information for the test scenario, such as PowerPack [11].

In other scenarios, power consumption information of a specific component (memory, disk, network, GPU) is needed. In those cases, *fine-grained measurements* are used. The implementation of the physical measurements included in this article for measuring CPU consumption was inspired in [9, 21] which, using Hall effect sensors, allows extracting the information from the corresponding device.

Other works have considered fine-grained measurements through sensors as invasive and highly complex, and have therefore studied estimation models based on hardware counters information. Such is the case of [4, 6, 7, 15, 17], where the power consumed by the entire system or its different components is estimated by means of fine-grained measurements. There are other models that are application-centered and whose objective is assessing application execution frequency [18, 5] or application scheduling [22]. More complex models have also been developed, such as models that include heterogeneity and consider GPU consumption [20, 24] and even models with a certain type of intelligence through *machine learning* [23].

Power estimation through fine grain models provided by processor manufacturers, such as *Running Average Power Limit* (RAPL) in the case of Intel [8, 14] and *Application Power Management* (APM) for AMD [1], have generated interest in the community in relation to whether the values obtained with them match actual physical measurements. For this reason, the reliability of counter registers and the consumption model have been questioned, and there are numerous works that attempt to validate the estimation by comparing the values obtained by the model with those obtained by the physical measurement methods mentioned above [9, 12, 13].

There are works that estimate coarse-grained power (or alternating current power of the entire system) from the fine-grained power model provided by processor manufacturers [16, 4].

3 Power consumption measurement

There are several approaches to measure the energy used during a certain period of time; however, only two of those are widely used.

On one hand, energy consumption can be monitored through certain *software measurement* interfaces that feed a refined power model with different system variables to obtain an estimated measurement of the current power consumption by the device. On Intel processors these variables are obtained through hardware counter records present in most processors designed in recent years, more specifically, since SandyBridge micro-architecture [14].

On the other hand, power consumption control through external *physical measurement* uses different tools that are especially designed for that. There are publications where the power of the entire system is sampled through a clamp probe connected to a digital oscilloscope [2, 3]. In other cases, invasive techniques are used, where current sensors (Hall Effect) are attached to the power sources of the various devices in a computer [11].

3.1 Software approach

For many years, microprocessor manufacturers installed hidden registers used for debugging hardware problems, but it wasn't until 1993 that their existence, and the possibilities they offer in relation to knowing very accurate data on processor status, became publicly known. Hardware counters are a set of special-purpose registers built in most modern micro-processors whose function is to store in a cumulative way certain events related to system hardware. In its latest designs, Intel has added specific events to measure power consumption.

Due to the potential offered by these counters, many tools have been developed that allow to specify events to be monitored, either for application profiling or to make adjustments during algorithm execution.

In the case of application profiling, *perf* emerges as a GNU/Linux application that displays a large number of events used in profiling and allows accessing them through the command line. Available events depend both on the architecture as well as on the Linux kernel being used. Due to the complexity involved in accessing counter values in real time, PAPI [19] turns up as an interface that allows accessing the various events and counter records from an algorithm's source code. Thus, it is possible to alter the application behavior based on the obtained power values.

As already mentioned, the design of consumption events led to research works focused on estimating power consumption for the different components in a computer system, obtaining power estimation models such as RAPL [8]. RAPL is a model that estimates energy using hardware counters and I/O models. RAPL uses measurement domains, which ensures fine-grained monitoring that may vary based on processor model.

The Package (PKG0) domain includes the power used by the entire socket. Then, PP0 encompasses the entire sub-system of cores and caches, while PP1 considers the consumption generated by circuitry that is external to the cores, such as onboard GPU. Lastly, the DRAM domain generates information about the energy used by the main memory. Since the PKG0 domain includes the PP0 and PP1 sub-domains, this is the domain that was selected to carry out energy measurements in this work.

3.2 Physical approach

When physically monitoring power, there are two procedures that can be carried out: a coarse-grained procedure and a fine-grained one. The former consists in intercepting, using a suitable instrument, the variables required for calculating instant power: current and voltage. In a fine-grained procedure, the instant current that goes through the power inlet of the main devices in the system is monitored.

This article focuses on the second procedure. To this end, several devices are used. First, an Arduino UNO development board based on free hardware and equipped with an ATmega328P processor, 14 digital I/Os, 6 analog I/Os and a 16MHz rate clock. Then, a couple of ACS712 sensors, which will be used to measure power through invasive Hall Effect. This type of sensors operate in a range between -5 and 5 Amps and have a resolution of 185mV/A, powered at 5V, as shown in Figure 1a. On the other hand, the equipment whose energy consumption will be measured should have a power source with a 4-pin ATX 12V connection (see Figure 1b), which is used mainly to power the CPU through the connector that can be seen in Figure 1b.

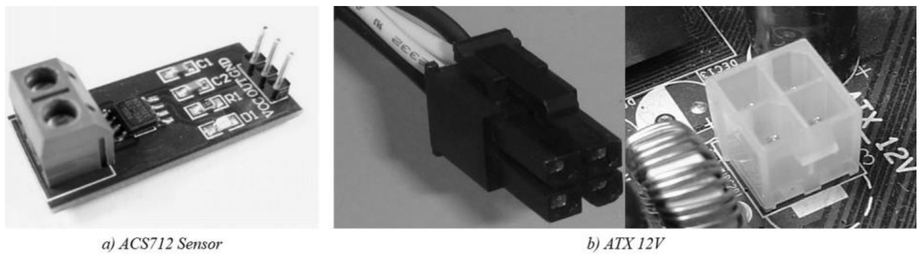

Fig. 1. *Devices required to carry out physical measurements.*

4 Implementation

To carry out the measurements, two machines are required – a *study machine* which runs the target algorithm, and a *monitoring machine* that allows to capture physical measurements.

Figure 2 shows a simple connection diagram. The mentioned sensors are serially connected to the ATX 12V power source, and the development board reads the values and send them to the monitoring machine. There, it runs a Python script responsible for receiving the data and save them to disk.

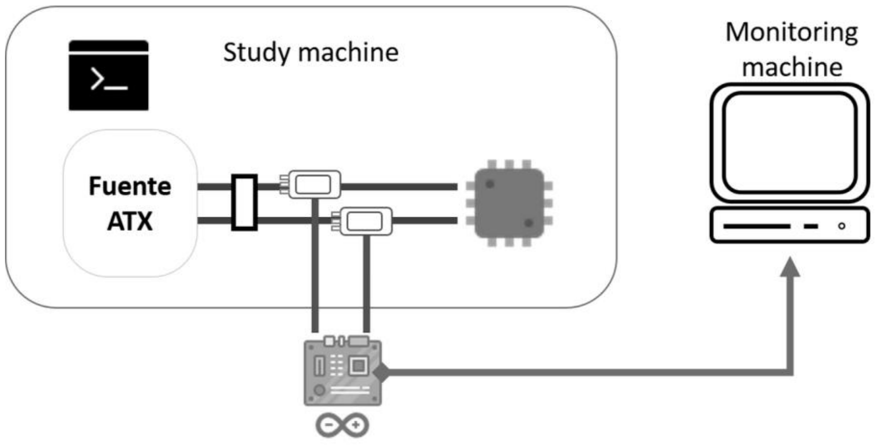

Fig. 2. *Measurement diagram.*

Finally, software-based measurements are carried out simultaneously with physical-based sampling using the *perf* tool. It should be noted that both procedures can be achieved at the same time because perf is a very light tool and, therefore, does not introduce a significant overhead. Synchronization between both machines is given through a TCP socket.

5 Experimentation

The specific measurements were performed on the machine labeled *study machine*, which consisted on a Gigabyte GA-H81M-H motherboard, an Intel Core I3-4170 processor (2 cores), an 8GB DDR3 RAM memory and Ubuntu 14.04 operating system (Kernel 3.16). The *monitoring machine* requires only one Python interpreter to obtain the values being sensored.

On the other hand, a basic matrix multiplication algorithm (1) was chosen as the case study because it represents a widely used problem which is part of solutions to larger-scale problems.

$$C_{i,j} = \sum_{k=1}^{p} A_{i,k} * B_{k,j} \qquad \textbf{(1)}$$

There exists two different implementations, a serial and a multithreaded one using the OpenMP library. Both of them at an operation rate of 1.6GHz.

6 Results

This section details the results corresponding to physical and software measurements. We include a brief graphical and descriptive analysis of the obtained data and then we discuss the results for the serial solution at different workloads. It should be noted that both physical and software measurements are done using an execution script that includes an idle-sleep segment (considered in the results) before and after the algorithm computation, as follows:

_> *Sleep 2s (Idle State)*
_> *Program execution*
_> *Sleep 2s (Idle State)*

Figure 3 shows the power consumption of the serial version with a 2048 x 2048 matrix.

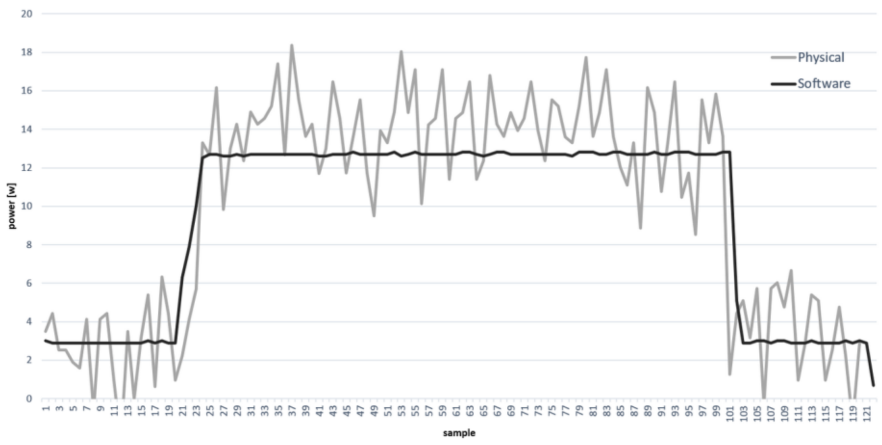

Fig. 3. *Consumption profile, serial algorithm.*

As it can be seen, the behavior of both approaches are similar. It should be noted that physical measurements appear to be "noisy" because we present instant data (i.e., not an average of sensored samples).

Below Table 1 lists average power values corresponding to both physical and software approximations when varying the workload for the serial algorithm.

Table 1. *Active power (Watts) for the serial solution.*

	1024	***2048***	***4096***	***8192***
Software measurement	12.35	12.70	13.07	12.90
Physical measurement	13.69	13.97	14.38	14.34

It is evident that the physical approach return higher values than software-based measurements. This deviation presents a linear behavior and shows a variation close to 10% (see Figure 4).

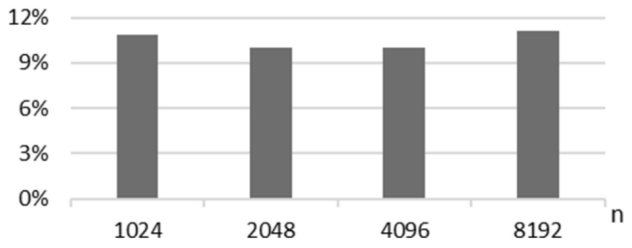

Fig. 4. *Error ratio for the serial implementation.*

On the other hand, OpenMP solutions using 2, 4 and 8 threads yield the results that are shown in Figure 5 (a, b and c, respectively).

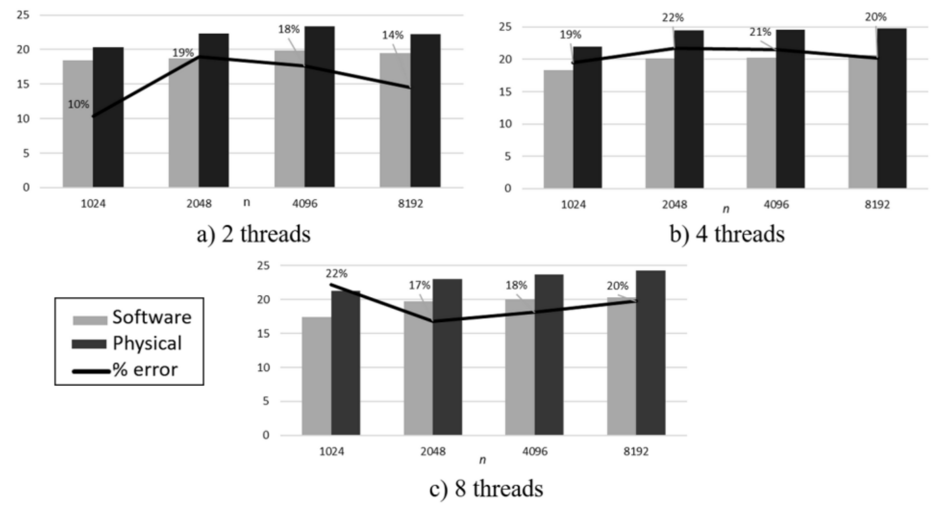

Fig. 5. *Average power consumption for thread-based solutions (in Watts). The offset between software and physical measurements is also shown.*

As shown in the graph, it can be concluded that there is an increase in the offset between both measurements and it ranges between 10% and 22%.

7 Conclusions and future work

This work focused on monitoring and analyzing energy consumption for a matrix multiplication application through physical measurements and a software interface based on hardware counters.

A physical measurement environment was built, which consisted of an Arduino board with current sensors that allowed recording real measurements of the average power demanded by the processor. In addition, a data-capturing script was implemented in Python to store power data. On the other hand, *perf* was used to access the values returned by RAPL for the Intel processor being studied.

Based on the results, a linear deviation in average power values was observed between physical and software-based measurements, the former being higher in all the cases that were assessed. This difference between measurements is believed to also be affected by one or several hardware devices whose software-recorded consumption has not been analyzed. Additionally, a variable error between 10% and up to 22% was detected in matrix multiplication experiments, for different data sizes and threads number.

Finally, in the future we plan to work on the detection and analysis of the gap between software-based and physical measurements, as well as on the correlation between the internal methods versus an external AC measurement. After the external measurement validation, there is great interest in assessing power consumption in other system components, such as RAM memory and I/O, which have not been considered in this work.

References

1. AMD, AMD Family 15th Processor BIOS and Kernel Developer Guide (2011)
2. Balladini J., Muresano R., Suppi R., Rexachs D., Luque E.: Methodology for predicting the energy consumption of SPMD application on virtualized environments. En Journal of Computer Science & Tecnology. pp. 130-136. 13. 3 (2013)
3. Balladini J., Rucci E., De Giusti A., Naiouf M., Suppi R., Rexachs D., Luque E.: Power Characterisation of Shared-Memory HPC Systems. In Computer Science & Tecnology Series. XVIII Argentine Congress of Computer Science Selected Papers. pp. 53-65 (2013)
4. Bircher W., John L.: Complete system power estimation using processor performance events. IEEE Tran. on Comp. 61. 4 (2012)
5. C. Lively, X. Wu, V. Taylor, S. Moore, H. Chang, C. Su, Cameron K.: Power-Aware Predictive Models of Hybrid (MPI/OpenMP) Scientific Applications on Multicore System. Computer Science – Res. and Dev. 27. 4 (2012)
6. Chen X., Xu C., Dick R., Mao Z.: Performance and Power Modeling in a Multi-Programmed Multi-Core Environment. In: Design Automation Conference (DAC), 2010 47th ACM/IEEE, pp 813-818 (DAC 2010)
7. Contreras G., Martonosi M.: Power Prediction for Intel XScale Processors Using Performance Monitoring Unit Events. In: Proceedings of the 2005 International Symposium on Low Power Electronic and Design, 2005. pp 221-226. ISLPED'05 (2005)
8. David H., Gorbatov E.: RAPL: Memory Power Estimation and Capping. In: 2010 ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED). pp. 189-194 (2010)
9. Desrochers S., Paradis C., Weaver V. M.: A validation of DRAM RAPL Power Measurements. In: Second International Symposium on Memory Systems 2016. pp 445-470.
10. Diouri M. E. M., Dolz M.: Assessing Power Monitoring Approaches for Energy and Power Analysis of Computers. Sustainable Computing: Informatics and Systems, Elsevier. pp. 68-82. 4. 2 (2014)
11. Ge R., Feng X., Song S., Chang H. C., Li D., K. W. Cameron: PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications. IEEE Transactions on Parallel and Distributed Systems. pp. 658-671. 21. 5 (2010)
12. Hackenberg D., Ilsche I., Schone R., Molka D., Schmidt M., Nagel W. E.: Power Measurement Techniques on Standard Compute Nodes: A Quantitative Comparison. IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS) (2013)
13. Hahnel M., Dobel B., Volp M., Hartigl H.: Measuring Energy Consumption for Short Code Paths Using RAPL. Technische Universität Dresden. ACM SIGMETRICS Performance Evaluation Review Volume 40. pp. 13-17 (2012)
14. Intel Architecture Software Developer´s Manual, volumen 3: System Programming Guide, (2009)
15. Isci C., Martonosi M.: Runtime Power Monitoring in High-end Processors: Methodology and Empirical Data. In: 36th IEEE/ACM International Symposium on Microarchitecture (2003). pp 93.
16. Khan K. N., Ou Z., Hirki M., Nurminen J. K., Niemi T.: How much power does your server consume? Estimating wall socket power using RAPL measurements. Computer Science - Research and Development. pp. 207-214 (2016)
17. Lim M., Porterfield A., Fowler R.: SoftPower: Finegrain Power Estimations Using Performance Counters. In: Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing. pp 308-311. (HPDC '10) (2010)

18. Lively C., Taylor V., Wu X., Chang H., Su C., Cameron K., Moore S., Terpstra D.: E-AMOM: An Energy-Aware Modeling and Optimization Methodology for Scientific Applications on Multicore Systems. Computer Science – Res. and Dev. 29 (3-4), pp 197-210 (2014).
19. Mucci P., Browne S., Deane C., Ho G.: PAPI: A Portable Interface to Hardware Performance Counters. In: Proceedings of the Department of Defense HPCMP Users Group Conference. pp. 7-10 (1999)
20. Nagasaka H., Maruyama N., Nukada A., Endo T., Matsuoka S.: Statistical Power Modeling of GPU Kernels Using Performance Counters. In: International Green Computing Conference, 2010. Pp 115-122
21. Picariello F., Rapuano S., Villano U.: Evaluation of Power Consumption of Workstation Computers using Benchmarking. In: Proceedings 12th IMEKO TC10 Workshop on Technical Diagnostics: New Perspective in Measurements, Tools and Techniques for Industrial Applications University of Sannio, Italy. pp. 242-247 (2013)
22. Singh K., Bhadhauria M.,McKee S. A.: Real Time Power Estimation and Thread Scheduling via Performance Counters. In: Workshop on Design, Architecture and Simulation of Chip Multi-Processors. Vol. 37, No 2, May 2009. pp 46-55
23. Song S., Su C., Rountree B., Cameron K.: A Simplified and Accurate Model of Power-Performance Efficiency on Emergent GPU Architectures. In: 2013 International Symposium on Parallel & Distributed Processing. pp 673-686. IPDPS2013
24. Tsafack Chetsa G.L., Lefevre L., Pierson J.M., Stolf P., Da Costa G.: Exploiting Performance Counters to Predict and Improve Energy Performance of HPC Systems. Future Gene. Computer Systems. 36 (2014)