ANN. MULT. GPU PROG.

Power and Energy Implications of the Number of Threads Used on the Intel Xeon Phi

O. G. Lorenzo[1*], T. F. Pena[1], J. C. Cabaleiro[1], J. C. Pichel[1], F. F. Rivera[1] and D. S. Nikolopoulos[2]

[*]Correspondence:
oscar.garcia@usc.es
[1]Centro de Investigación en Tecnoloxías da Información (CITIUS), Univ. of Santiago de Compostela,Santiago de Compostela, Spain
Full list of author information is available at the end of the article

Abstract

Energy consumption has become an important area of research of late. With the advent of new manycore processors, situations have arisen where not all the processors need to be active to reach an optimal relation between performance and energy usage. In this paper, a study of the power and energy usage of a series of benchmarks, the PARSEC and the SPLASH-2X Benchmark Suites, on the Intel Xeon Phi for different threads configurations, is presented. To carry out this study, a tool was designed to monitor and record the power usage in real time during execution time and afterwards to compare the results of executions with different number of parallel threads.

Keywords: Power; energy; manycores; Intel Xeon Phi; benchmarking

1 Introduction

Manycore systems have been hailed as a important step towards a greater energy efficiency. They increase the computer performance through replicating simple and energy conserving cores on a single chip and also promise to reduce energy usage by allowing resources to be used only when necessary. In these systems, optimally allocation of the available power budget to different parts of the system is becoming a crucial decision. In fact, different computations will suffer different kind of performance degradation when power assigned to the components is modified. Therefore, there exists a need for analysis and tools that help the user to understand how a given computation behaves in terms of power usage.

Intel Xeon Phi is the first commercial manycore x86-based processor, currently available as an accelerator to be used alongside a host system. The Xeon Phi attempts to lower the energy cost of computation by favoring a high degree of parallelism over single core performance. One of the main advantage of the Xeon Phi over other accelerators as Graphics Processing Units (GPUs) or Field Programmable Gate Arrays (FPGAs), is its compatibility with x86 instructions, which allows the same code to be run on either the host processors or the accelerator board.

For the Xeon Phi, some models have been proposed to improve energy efficiency at core or instruction level [1], and they are certainly useful for new codes or to find the best implementations of legacy ones. Nevertheless, few codes designed to be run on regular SMP machines scale easily to the number of threads available on the Xeon Phi and, if scalability is limited, using all the resources on the Xeon Phi may be detrimental to energy consumption. Since as more cores are used, and more intensely, more power is consumed. Therefore, energy consumption may reach its minimum using a different number of threads for different applications. There may be codes which consume less energy using fewer threads, with the Xeon Phi using less power, even when their execution time may be longer. For these reasons it is interesting to find the best thread configuration (in number and also in placement) for legacy applications.

In this paper a study of the power and energy usage of a series of benchmarks, the Princeton Application Repository for Shared-Memory Computers (PARSEC) and the Stanford ParalleL Applications for SHared-memory 2 (SPLASH-2X) Benchmark Suites [2], on the Xeon Phi for different number of threads involved, is presented. To carry out this study a set of tools were designed to monitor and record the power usage in real time during the execution of the codes and afterwards compare the results of different executions with different number of parallel threads.

This study shows that finding the best configuration in terms of number of threads, either from the performance or the energy efficiency point of view, is far from being straightforward. The optimal balance between performance and energy is not easily reached, and using all the available hardware resources may not yield the best performance or energy usage.

The rest of the paper is organised as follows: Section II reviews some related work. Section III describes the Intel Xeon Phi architecture. The mechanisms for power measurement on the Xeon Phi and the tool we have developed to facilitate the obtaining of power data are introduced in Section IV. In Section V some of the results of our analysis of the above mentioned benchmarks are presented. Finally, some conclusions are drawn in Section VI.

2 Related work

Gaining a good understanding of the power and energy usage behaviour of HPC codes is essential, given that the power wall has emerged as the key bottleneck in the design of exascale systems. In [3], a method to obtain this behaviour in a detailed way on a manycore system is shown. This method is integrated in a tool that interacts with the user and shows graphically some measurements.

There are several instrumentation systems available [4, 5] that supply users with power and energy consumption information. However, many suffer from resolution and accuracy problems. Moreover, some of them supply results out-of-band, impeding optimization strategies to improve energy consumption.

Modeling power and energy is a growing topic in HPC computing, many contributions can be found on this issue [6, 7, 8]. Models to estimate leakage energy on a cache hierarchy are presented in [9]. In [10], the authors present a detailed study of the performance-energy tradeoffs of the Xeon Phi architecture. Leon and Karlin [11] demonstrate key tradeoffs among power, energy and execution time for explicit hydrodynamics codes. In [12], a model to predict the performance effects of applying multiple techniques simultaneously is presented. In [13], models for predicting CPU and DIMM power and energy were introduced.

3 The Intel Xeon Phi

The architecture of the Intel® Xeon Phi™ is referred to as Intel Many Integrated Core Architecture or Intel MIC. It is a coprocessor computer architecture developed by Intel incorporating earlier work on the Larrabee [14] many core architecture, the Teraflops Research Chip [15] multicore chip research project and the Intel Single-chip Cloud Computer[16] multicore microprocessor. The cores of Intel MIC are based on a modified version of P54C design, used in the original Pentium. The basis of the Intel MIC architecture is to leverage x86 legacy by creating a x86-compatible multiprocessor architecture that can utilize existing parallelisation software tools. Design elements inherited from the Larrabee project include x86 ISA, 4-way SMT per core, 512-bit SIMD units, 32 KB L1 instruction cache, 32 KB L1 data cache, coherent L2 cache (512 KB per core), and ultra-wide ring bus connecting processors and memory (see Figure 1). Manufactured using Intel industry-leading 22 nm technology with 3-D Tri-Gate transistors, each coprocessor features more cores, more threads, and wider vector execution units than an Intel Xeon processor. The high degree of parallelism is intended to compensate for the

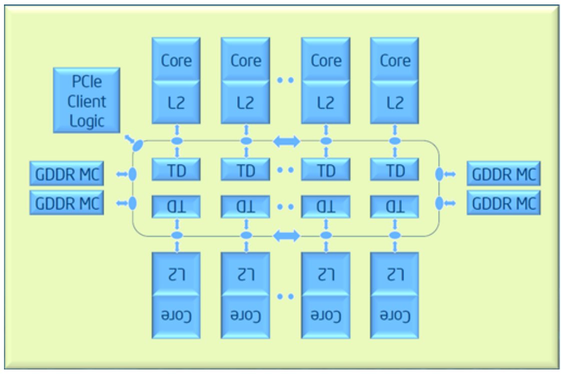


Figure 1 PHI memory architecture.

lower speed of each core to deliver higher aggregate performance for highly parallel workloads. Since languages, tools, and applications are compatible for both Intel Xeon processor and Intel Xeon Phi coprocessors, codes initially developed for Intel Xeon can be reused.

With Xeon Phi, a single programming model can be used for all the code. The coprocessor gives developers a hardware design optimized for extreme parallelism, without requiring them to re-architect or rewrite their code. Coming from Xeon programming there is no need to rethink the entire problem or learn a new programming model; existing codes can simply be recompiled and optimised using familiar tools, libraries, and runtimes.

Also, by maintaining a single source code between Intel Xeon processors and Intel Xeon Phi coprocessors, developers should be able to optimise once for parallelism but maximize performance on both processor and coprocessor.

While designed for high-performance computing, the coprocessor can host an operating system, be fully IP addressable, and support standards such as Message Passing Interface (MPI), unlike a GPU. This means it can operate in multiple execution modes. Workload tasks can be shared between the host processor and coprocessor, they can work independently or parts of the workload can be sent out to the coprocessor as needed (as in a GPU).

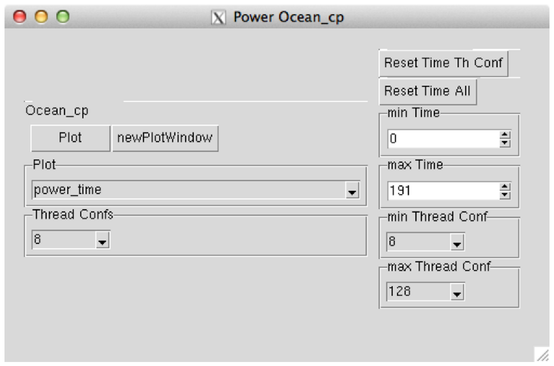
4 Power Measurement

4.1 Power Measurement on Xeon Phi

Power measurements for the Intel Xeon Phi coprocessor can be obtained directly from the Operating System, OS. There are no readable PM PMU events (Power Management Performance Monitoring Unit events) in the current first generation of Intel Xeon Phi coprocessors, although this is likely to change in later generations. The OS gets its temperature readings from off chip monitoring sensors on the circuit board supporting the processor. As such, it is an aggregate and approximate measurement.

When used in native mode, that is a user is directly logged in the coprocessor OS, power measurements can be read in `/sys/class/micras/power`. From the host system this measures are read using a command line tool provided by Intel [17]. These measurements are updated each 50 ms. The exported data consists in a series of values of power readings from various sensors. The coprocessor board sensors give information about power consumption. Sensors in the board measure the power from the various power inputs in uW. They are the following:

- Total power: Two measurements are given, each taken in different time windows (0 and 1).
- Instantaneous power and maximum instantaneous power.



Ocean_cp
Plot: power_time
Thread Confs: 8
min Time: 0
max Time: 191
min Thread Conf: 8
max Thread Conf: 128

Figure 2 The R GUI.

- PCI-E connector power (up to 75 W).
- Power from the auxiliary 2x3 (75 W) and 2x4 (150 W) connectors. These are needed because the Xeon Phi board can not be powered only by the PCI-E. (Some models may not need the 2x3 connector.)

Sensors near the Phi coprocessor also give information about current (in uA) and voltage (uV). These sensors are:

- Core rail. This sensor measures the activity of the cores. As more is demanded from the cores larger will be the power used.
- Uncore rail. This sensor measures the uncore activity, such as the bus ring interconnection among cores and the L2 cache level.
- Memory subsystem rail. This sensor measures the activity of the memory modules and their connection to the processor. While memory and cores are connected by the same ring bus, memory modules reside outside the processor chip.

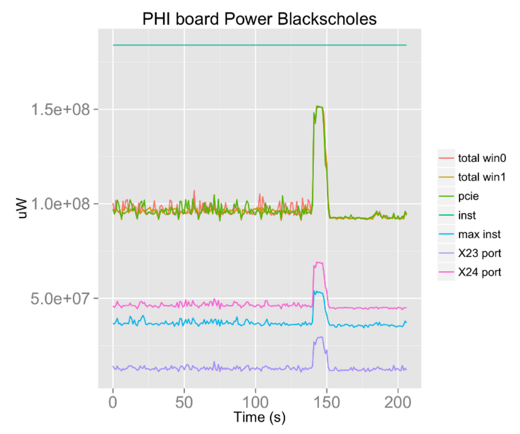
In addition to power, temperature measurements are also recorded, from eight different temperature sensors.

4.2 Power Measurement Tool

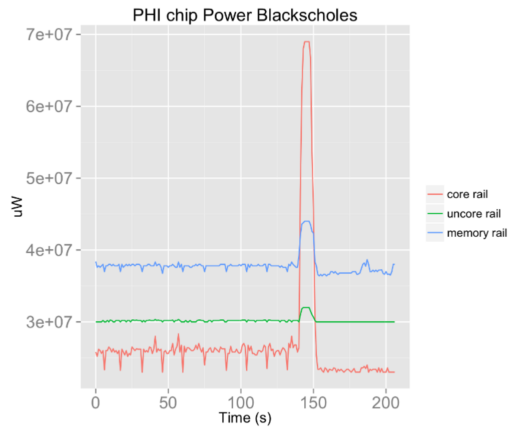
To record the power consumption of a benchmark, we have implemented series of bash scripts, which record to a file the commands used, the output of the benchmark and the power data. These scripts can be used to easily run a series of executions of an application with different thread configurations. To batch process these power data files, we have also implemented an application running on an R environment [18]. This tool mean averages the results of each different run of each thread configuration, to obtain a more representative view of its execution. Moreover, it allows to easily make comparisons among different thread configurations. A view of this interactive Graphical User Interface (GUI) is shown in Figure 2. In the left hand side of the GUI are the controls to select the kind of graph to plot, for instance power vs. time or energy vs. thread configuration, and the desired thread configuration (if necessary). In the right hand side are the controls to modify the limits of the x axis of the graph (time or thread configuration) and buttons to reset those limits to the default ones.

5 Case Study

Power and energy consumption of a series of benchmarks executed natively on the Xeon Phi are considered. Benchmarks were executed with different number of threads, and during each execution power measurements were taken every second. The results for 5 different executions of each thread



(a) Power vs Time on board.



(b) Power vs Time on chip.

Figure 3 Power vs Time for Blackscholes benchmark.

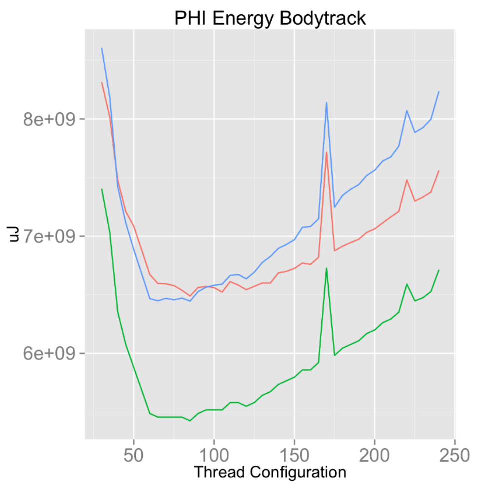
configuration were combined to obtain an average of the power usage. This way the evolution in time of the power usage can be extracted.

In addition, the energy consumption of each thread configuration can be studied. As more threads are executed more instantaneous power is used, but applications usually run faster, which means that total energy consumption may be lower, although this is not always the case.

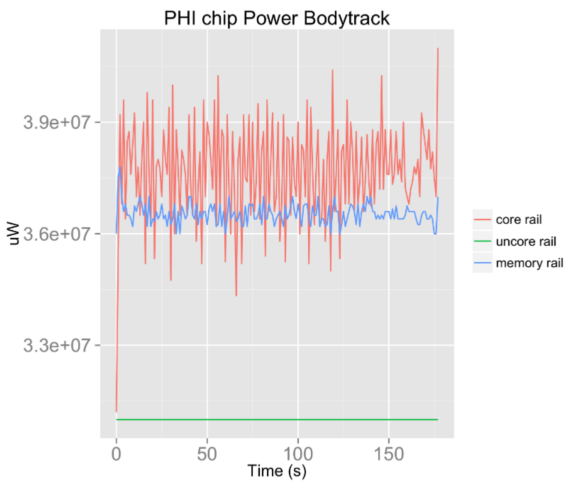
All benchmarks were compiled with `icc -O2` and autovectorisation, at least. The Xeon Phi coprocessor used was 7120A model, with 16 GB of memory and 61 cores at 1.238 GHz. In this coprocessor, up to 240 threads can be used to run applications, with 4 threads per core (one core is reserved for the OS). Thread placement was left to the OS, no directions were given, so threads were assigned in a round robin fashion. Alternative placement of threads, using affinity options, were not considered in this study. This means that, when up to 60 threads are considered, just one thread is executed per core. Whereas from 61 threads up, cores are assigned more threads, which implies that in some configurations there may be some cores executing one more thread than others. In this Xeon Phi board the data storage is implemented as Network Attached Storage (NAS) when programs are executed directly by the Phi.

5.1 Benchmark Suite

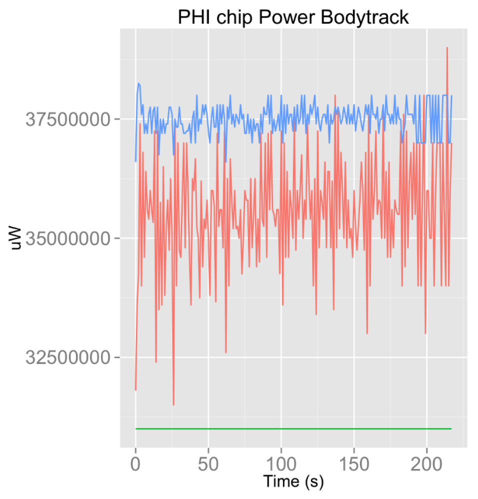
The benchmarks used to carry out this study were the PARSEC and SPLASH-2X benchmark suites [2]. PARSEC is a benchmark suite composed of multithreaded programs. The suite focuses on emerging workloads and was designed to be representative of next-generation shared-memory programs for chip-multiprocessors. It is complemented by the SPLASH-2X benchmarks, which is a benchmark suite that includes applications and kernels mostly in the area of high performance computing. It has been widely used to evaluate multiprocessors and their designs for the past 15 years. SPLASH-2X and PARSEC benchmark suites complement each other in terms of diversity of architectural characteristics such as instruction distribution, cache miss rate and working set size. In this study benchmarks were executed using their Native input parameters, the larger standard inputs, to obtain long execution times and be able to appreciate variations in the energy usage. Many of these benchmarks suffer a performance loss if compared to their execution on the Xeon host. This is mainly due to I/O, since the host can access the hard drive directly, but the coprocessor must use NAS when used independently.



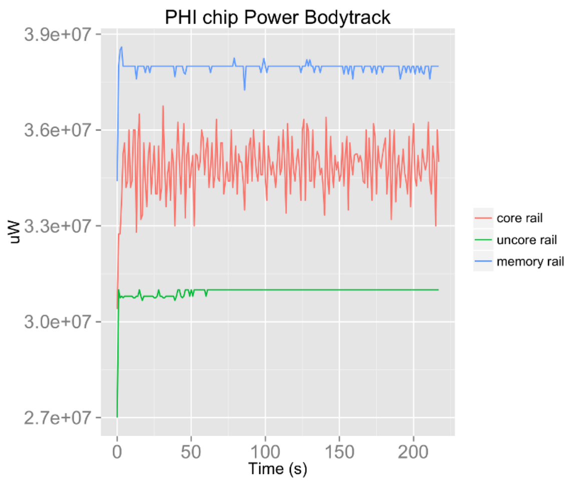
(a) Energy vs thread configuration.



(b) Power vs time 60 threads.



(c) Power vs time 170 threads.

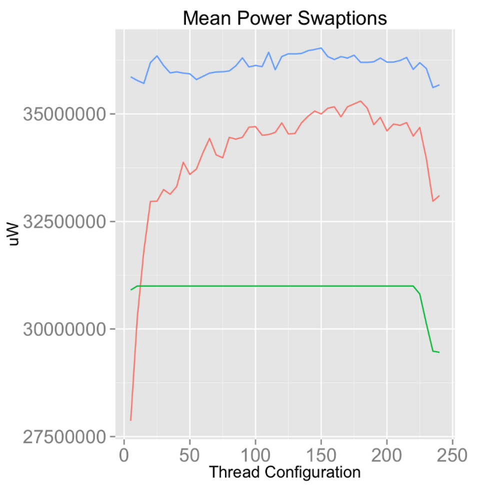


(d) Power vs time 240 threads.

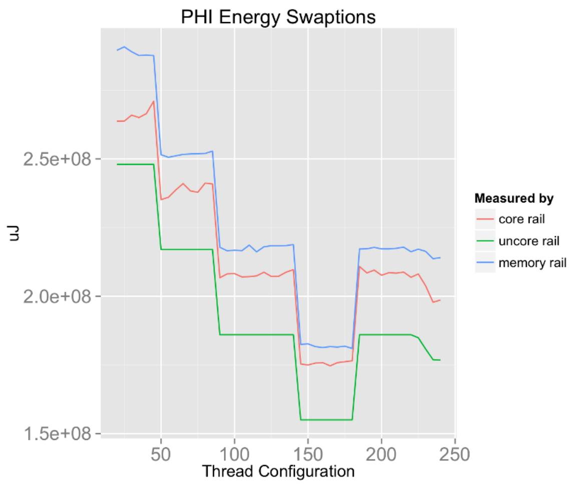
Figure 4 Power data for Bodytrack benchmark.

5.2 Data Analysis

First, note that there is a great correlation between the data power usage measured in the board sensors to those near the processor. For instance, consider the evolution of power usage during the execution of the Blackscholes benchmark (PARSEC) shown in Figure 3. It is clear how the peaks of power consumption in the processor are met with peaks on the board. It can be also observed how the 2x3 connector (only 75 W) draws less power than the 2x4 connector (150 W), and they seem to work in tandem, drawing each a proportional amount of the total power. Adding the power of these two connectors and the instantaneous power drawn from the PCI-E gives approximately the total power measured during window 1 and window 2 (Figure 3(a)). Furthermore, the total power (approximately 100 W during the initial phase, Figure 3(a)) is close to the sum of the core, uncore and memory rails power (approximately 25 W, 30 W and 37 W during the initial phase, in Figure 3(b)). This result is reasonable given that they do not comprise the entirety of the elements on the board. The fact that power sensors behave this way seems to indicate that their measurements can be trusted, at least in terms relative to one another. Given this correlation, in order to make figures more readable, only



(a) Mean power vs thread configuration.



(b) Energy vs thread configuration.

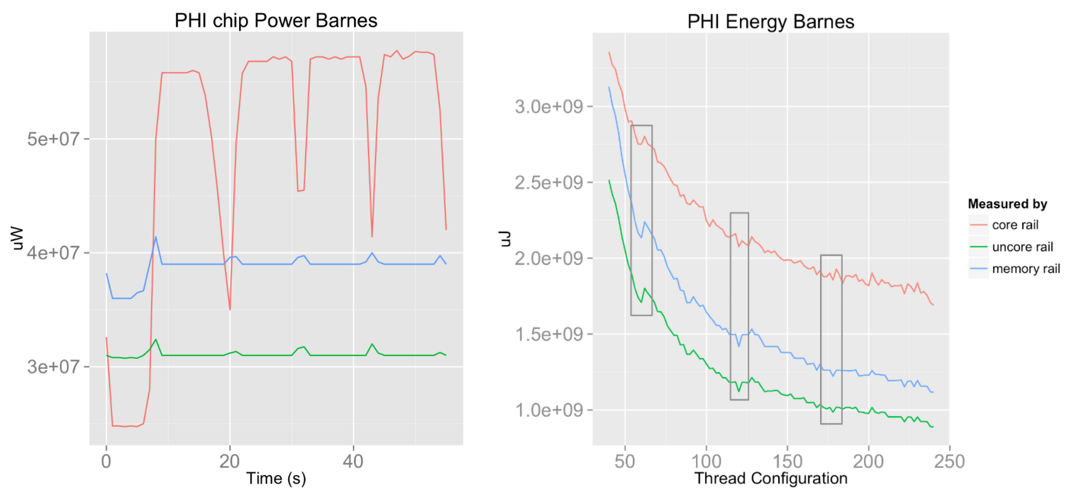
Figure 5 Power and Energy data for Swaptions benchmark.

data for the measurements of the processor rails will be shown in the remaining of the paper, since they are more interesting for our analysis.

Applications for the Xeon Phi are recommended to be executed with 4 threads per core. However, this configuration may not give the best performance, specially if the application is not modified from its Xeon version for the Phi. As an example of an application which performs better with 1 thread per core is Bodytrack (PARSEC). Bodytrack is a benchmark based on a body tracking application, which reads a series of images from disc as it were video. It performs badly on this Xeon Phi due to the high I/O due to necessity of loading the images from NAS. Furthermore, its best performance is when 60 threads are used, one thread per core, and gets worse as more threads are used, which means it consumes more energy (see Figure 4(a)). In Figure 4 variability on the power used by the cores can be seen. In Figure 4(b) the usage by the memory and uncore is fairly constant, and similar to the case in Figure 3(b), but the cores consumption varies greatly. Figure 4(d) shows Bodytrack executed with 240 threads. Note that how the cores, while now executing 4 threads each, actually use less power than in the case when it is executed with 60 threads, probably because they are performing less operations per second. A reason for this may be glimpsed in Figure 4(c), Bodytrack executed with 170 threads, one of the worst cases; here the memory is being stressed and cores are probably starved of data. It is likely data can not be well partitioned among the threads.

The Swaptions benchmark (PARSEC) performs well on the Xeon Phi. Its power requirements are very similar in any thread configuration, since it does not stress the cores (see Figure 5(a)). This leads to an energy consumption directly related to the execution time. The stair pattern in Figure 5(b) is due to the resolution of the measurement; because they are taken every second, and execution time is between 6 and 9 seconds, so detail is lost. Nevertheless, it is enough to see the best configurations are those that use 2 threads per core (between 120 and 180 threads).

In Figure 6, the behaviour of the benchmark Barnes, from the SPLASH-2X suite, shows that some interesting facts about an application execution can be deduced from its power usage. In Figure 6(a), it seems clear that the application goes through 4 phases of intense computation. Between those phases the memory and uncore rails flare up, possibly indicating a data preload or store. In Figure 6(b), four sections can be differentiated, each corresponding to cores using 1, 2, 3 or 4 threads; this behaviour



(a) Power vs time for 60 threads configuration.

(b) Energy vs thread configuration.

Figure 6 Power and Energy data for Barnes benchmark.

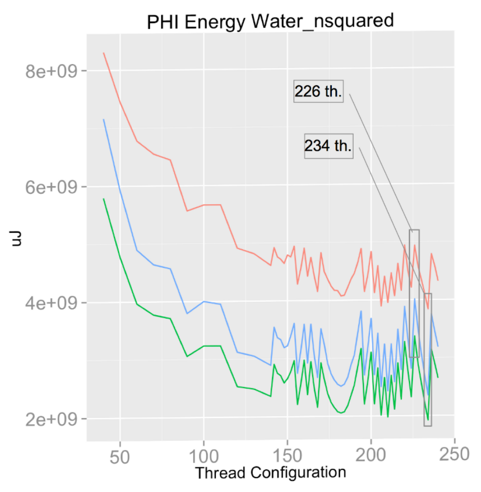
can be observed in other benchmarks, but it is clearer in this one. These sections are separated by a low energy peak (highlighted in the figure by a rectangle), that corresponds to multiples of the number of cores, that are 60, 120 and 180. Usually these configurations produce better results, since they share the workload more evenly, because they imply the same number of threads per core.

Nevertheless, these configurations may not always be the best, neither in execution time nor energy consumption. With the Water_nsquared benchmark (SPLASH-2x), shown in Figure 7, it can be observed how differences of just a few threads can influence greatly the execution, and how the best thread configurations are not always obvious. In Figure 7(a), this variability, which mostly follows the execution time, is clear. Note in Figure 7(b), how the mean power usage per second is almost the inverse of the total energy consumed. This is due to the fastest thread configurations drawing more power during execution, but using less energy in the end by finishing the execution faster. The two thread configurations highlighted on these figures are shown in more detail in Figures 7(c) and 7(d). In Figure 7(c), the phases of the execution are shown for one of the worst cases with 4 threads per core. Contrary to what Figure 6(a) showed, in which computations and memory accesses seem to occur at the same time. If Figure 7(c) and Figure 7(d), respectively, a bad and a good case are compared. It is clear how, with good thread configurations, power usage remains high during the whole execution. This leads to a greater energy consumption per second, but it is finally compensated by a lower execution time.

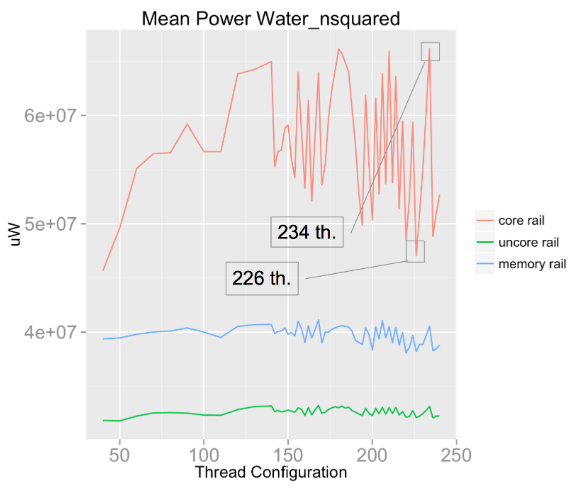
In Figure 8, a case presenting a long initialisation phase in just one core can be observed. Anyway, the compute phase is still long enough, so a performance gain using more cores can be appreciated energy wise. Still, the actual best thread configuration is not easy to choose, since a few threads up or down make a large difference in execution time.

A case where the energy usage remains similar in any configuration can be seen in Figure 9. In this example, due to a long initialisation phase on just one core, the performance gain in the compute phase with many cores is not enough to lead to energy savings.

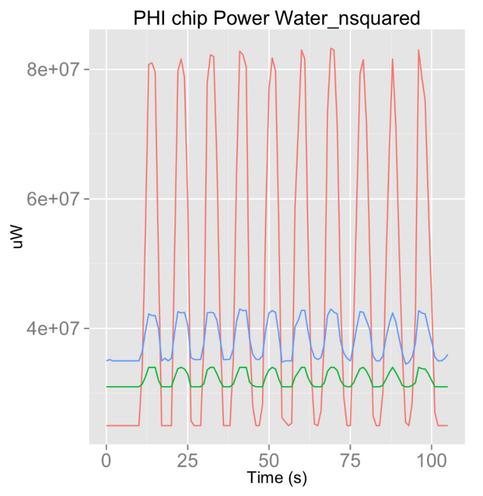
In general power consumption and performance seem to be highly related on the Xeon Phi. This leads to the best thread configuration from a performance view usually being the less energy consuming. Nevertheless, the benchmarks considered in this study perform worse on the Xeon Phi than on the



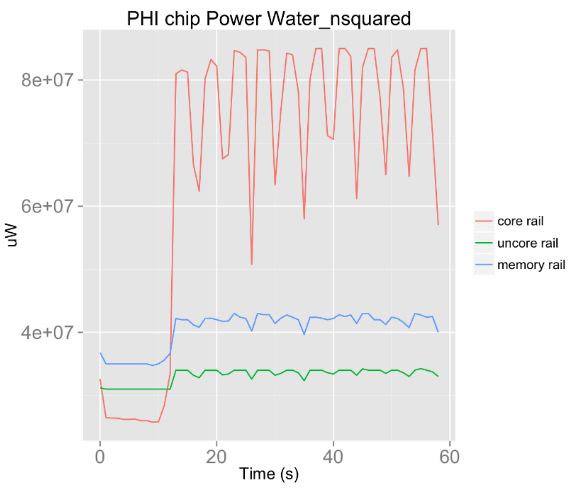
(a) Energy vs configuration.



(b) Mean Power vs configuration.



(c) Power vs vs time for 226 threads.



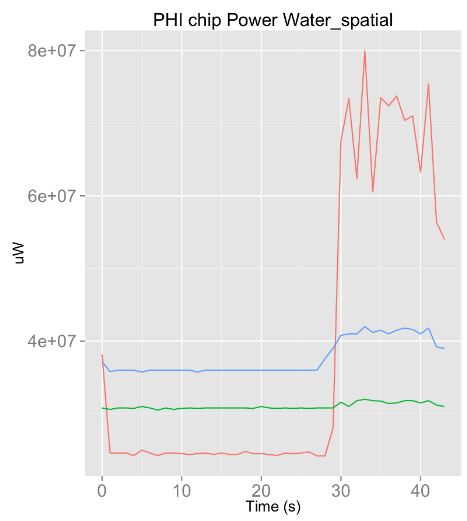
(d) Power vs vs time for 234 threads.

Figure 7 Power and Energy data for Water_nsquared benchmark.

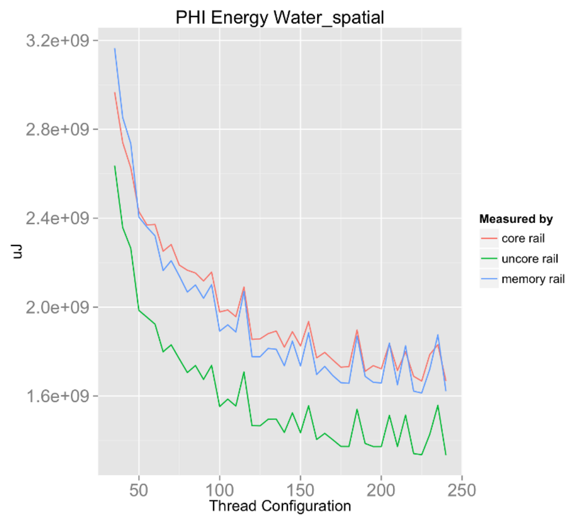
host system, a dual Xeon with 16 cores in total, sometimes in the order of 4 or 5 times worse, like Blackscholes. Some loss of performance is due to the NAS I/O, but other considerations, as the level of vectorisation, may be of importance. For instance, Xeon Phi cores allow a greater vectorisation than the usual Xeon cores, and the autovectorisation done by the `icc` compiler may not be enough to obtain good performance on the Xeon Phi.

6 Conclusion

Power and energy usage has become an important issue in all areas of computer science and information technology of late, particularly in HPC. The use of hardware accelerators such as the Xeon Phi may help to reach a better performance per watt ratio. Even so, the optimal balance between performance and energy is not easily reached. To use all the available hardware resources in order to obtain the maximum performance may not compensate the energy consumed, although in the Xeon Phi it seems the best starting point. Nevertheless, the best thread configuration, either from the performance or the energy efficiency point of view, may not be evident at first glance. In future work,

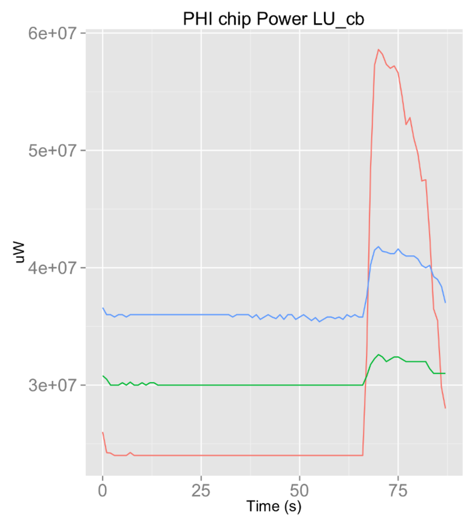


(a) water_spatial Power/time 240 threads.

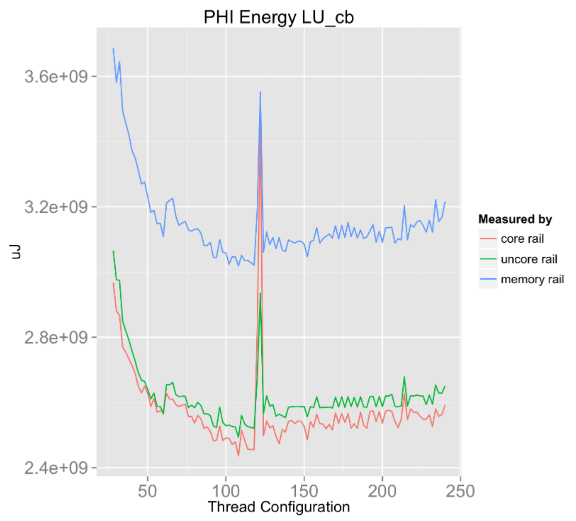


(b) water_spatial Energy/configuration.

Figure 8 Power and Energy data for water_spatial benchmark.



(a) lu_cb Power/time 128 threads.



(b) lu_cb Energy/configuration.

Figure 9 Power and Energy data for lu_cb benchmark.

a study of different configurations, changing not only the number of threads, but their core placement and affinity, may yield complementary insights. To help finding the best thread configuration for a given problem, a series of tools are here presented. These tools allow for a fast study of an application performance and power usage using different number of threads on the Xeon Phi. These tools could be improved in the future using data gathered from hardware counters, to try relating power usage to other performance metrics, as cache misses, for instance, or to gain a more detailed view of the power consumption, at core level or related to different execution phases of an application.

Acknowledgements
This work has been partially supported by the Ministry of Education and Science of Spain, FEDER funds under contract TIN 2013-41129P, and Xunta de Galicia, GRC2014/008 and EM2013/041. It has been developed in the framework of the European network HiPEAC-2, the Spanish network CAPAP-H, and Galician network under the Consolidation Program of Competitive Research Units (Network ref. R2014/041). This work is a result of a collaboration between CITIUS, University of Santiago de Compostela, and the School of Electronics, Electrical Engineering and Computer Science, Queens University Belfast.

Author details
[1]Centro de Investigación en Tecnoloxías da Información (CITIUS), Univ. of Santiago de Compostela,Santiago de Compostela, Spain.
[2]Queens University Belfast, Belfast, Northern Ireland, UK.

References
1. Shao, Y.S., Brooks, D.: Energy characterization and instruction-level energy model of Intel's Xeon Phi processor. In: Proceedings of the International Symposium on Low Power Electronics and Design, pp. 389–394 (2013). IEEE Press
2. Bienia, C.: Benchmarking modern multiprocessors. PhD thesis, Princeton University (January 2011)
3. Kogge, P., et al.: Exascale computing study: Technology challenges in achieving exascale systems. Technical report, DARPA (2008)
4. Bedard, D., Lim, M.Y., Fowler, R., Porterfield, A.: Powermon: Fine-grained and integrated power monitoring for commodity computer systems. In: IEEE SoutheastCon 2010 (SoutheastCon), Proceedings of The, pp. 479–484 (2010). IEEE
5. Tolentino, M., Cameron, K.W.: The optimist, the pessimist, and the global race to exascale in 20 megawatts. Computer **45**(1), 95–97 (2012)
6. Kestor, G., Gioiosa, R., Kerbyson, D.J., Hoisie, A.: Enabling accurate power profiling of HPC applications on exascale systems. In: ACM International Workshop on Runtime and Operating Systems for Supercomputers (2013)
7. Isci, C., Martonosi, M.: Runtime power monitoring in high-end processors: Methodology and empirical data. In: IEEE International Symposium on Microarchitecture, MICRO–36 (2003)
8. Bertran, R., Gonzalez, M., Martorell, X., Navarro, N., Ayguade, E.: Decomposable and responsive power models for multicore processors using performance counters. In: ACM International Conference on Supercomputing (2010)
9. Deshpande, A., Draper, J.: Leakage energy estimates for HPC applications. In: ACM International Workshop on Energy Efficient Supercomputing. E2SC'13 (2013)
10. Li, B., Chang, H.-C., Song, S., Su, C.-Y., Meyer, T., Mooring, J., Cameron, K.W.: The power-performance tradeoffs of the Intel Xeon Phi on HPC applications. In: Proc. 2014 IEEE Int. Parallel and Distributed Processing Symposium Workshops (IPDPSW'14), pp. 1448–1456 (2014)
11. Leon, E.A., Karlin, I.: Characterizing the impact of program optimizations on power and energy for explicit hydrodynamics. In: IEEE 28th International Parallel and Distributed Processing Symposium Workshops, pp. 773–781 (2014)
12. Nikolopoulos, D., Vandierendonck, H., Bellas, N., Antonopoulos, C., Lalis, S., Karakonstantis, G., Burg, A., Naumann, U.: Energy efficiency through significance-based computing. Computer **47**(7), 82–85 (2014)
13. Tiwari, A., Laurenzano, M.A., Carrington, L., Snavely, A.: Modeling power and energy usage of HPC kernels. In: IEEE 26th Int. Parallel and Distributed Processing Symp. Workshops, pp. 990–998 (2012)
14. Seiler, L., Carmean, D., Sprangle, E., Forsyth, T., Abrash, M., Dubey, P., Junkins, S., Lake, A., Sugerman, J., Cavin, R., Espasa, R., Grochowski, E., Juan, T., Hanrahan, P.: Larrabee: A many-core x86 architecture for visual computing. ACM Trans. Graph. **27**(3), 18–11815 (2008)
15. Vangal, S., Howard, J., Ruhl, G., Dighe, S., Wilson, H., Tschanz, J., Finan, D., Iyer, P., Singh, A., Jacob, T., Jain, S., Venkataraman, S., Hoskote, Y., Borkar, N.: An 80-tile 1.28 TFLOPS network-on-chip in 65 nm CMOS. In: IEEE International Solid-State Circuits Conference. Digest of Technical Papers, pp. 98–99. IEEE, San Francisco, USA (2007)
16. Rattner, J.: Single-chip Cloud Computer: An Experimental Many-core Processor from Intel Labs. http://download.intel.com/pressroom/pdf/rockcreek/SCC_Announcement_JustinRattner.pdf
17. Intel Xeon Phi Coprocessor Datasheet. http://www.intel.com/content/www/us/en/processors/xeon/xeon-phi-coprocessor-datasheet.html
18. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2008). R Foundation for Statistical Computing. ISBN 3-900051-07-0