Quantifying the Dominance of Leakage Energy in Large-Scale System Caches

Aditya Deshpande and Jeffrey Draper
Information Sciences Institute
University of Southern California
Los Angeles, CA, USA
{adeshpan, draper}@isi.edu

ABSTRACT

Energy consumption is becoming a critical metric in the design and use of large-scale high-performance systems. With large on-chip caches and advances in chip fabrication technologies, on-chip caches account for a large proportion of total leakage power losses. In this work, we quantify on-chip cache leakage-power losses across a wide set of parallel applications and compare the leakage and dynamic energy consumption for various levels of on-chip caches. Our scheme profiles an application to measure cache accesses in order to estimate the energy consumption for various levels of caches. Our study indicates that for various levels of cache, leakage-energy consumption significantly dominates dynamic-energy consumption, and this trend of leakage domination in on-chip caches is expected to increase with every new generation of semiconductor process. Our study indicates that the problem of leakage in on-chip caches cannot be neglected in attacking the energy barrier for building extreme-scale systems.

Categories and Subject Descriptors

C.4 [**PERFORMANCE OF SYSTEMS**]: Design studies; J.0 [**COMPUTER APPLICATION**]: General

General Terms

Measurement, Cache leakage

Keywords

energy estimation, leakage energy, cache energy, HPC applications

1. INTRODUCTION

Energy consumption in large-scale high-performance systems has become a major concern due to rising operational costs and scalability issues. For example, Titan [1] consumes over 8MW of power for delivering over 20PFlops of performance. The electricity costs alone associated with operating Titan or other similarly massively parallel systems are very high. With the next generation of supercomputing machines expected to usher in an "exa-scale" era of computing, increasing operational costs would become a hindrance to their deployment. With $50x$ performance improvement needed beyond current state-of-the-art machines to reach exa-scale computing, a corresponding $50x$ increase in the energy requirements would make it impossible to operate these machines. This energy challenge in high-performance systems has been identified as a key barrier to exa-scale systems [2].

Regardless of programming model used for these large-scale high-performance systems, very little information is available to application developers regarding the energy consumption of their applications. Energy consumed during application execution can be divided into various components, for example, core, memory, network and peripheral I/O's, etc. The energy usage levels for all these components in the system remain hidden from the application developer during application execution. If the energy consumption for various components is made available to application developers, it could provide insight into where exactly energy is needlessly wasted in an application execution. In the best case, changes to the algorithm/application could then be made to reduce the energy footprint. Profiling applications for energy consumption on current high-performance systems is an important step in identifying and understanding energy bottlenecks in current systems and would allow extreme-scale system developers to avoid these bottlenecks in future designs.

Prior research [2] has identified the memory subsystem as one of the most critical blocks from an energy perspective. While prior work has developed a coarse-grain understanding of memory energy requirements from a system perspective, there is little information on how various levels of the memory hierarchy interact with each other and their energy implications from an application execution view. More detailed studies of energy consumption in memory systems are needed to enable a better understanding of the relationship between energy profiles and contributing factors. Also, with computer architects primarily focusing on performance, the amount of on-chip caches in the system has continued to increase. Such large on-chip caches occupy a large portion of total chip area and also account for up to 50% of total chip power budgets. These trends further reinforce the need for more detailed studies of memory system energy consumption.

As chip fabrication technology has progressed with minimum transistor size decreasing, the amount of leakage or static energy loss has increased dramatically [3]. With every generation of fabrication technology, the ratio of leakage energy contribution to total chip energy has increased, leading to higher energy budgets. As on-chip caches account for a significant portion of total chip area and therefore total leakage, they make a very good target for designers to evaluate the impact of leakage energy consumption in various levels of caches during an application execution. Current high-performance systems have enormous aggregated on-chip caches, and even a small reduction in the energy consumed in these caches would have a profound impact on the overall energy consumption of a system. In this work, we focus our attention on the distribution of energy in on-chip caches, and we quantify and compare the dynamic versus leakage energy dissipation. Here, we present data for L1 data/instruction caches. As they are the smallest in area and most often accessed caches, we expect them to have the lowest percentage of leakage energy in all of the cache hierarchy.

2. METHODOLOGY

We profiled application execution with regard to cache activity to compute energy values. In our profiling scheme, we instrument application executables to measure hardware performance counter information (PAPI counters) [4] related to cache activity. The recorded counter values are then combined with the cache access energy information obtained from the cache energy modeling tool CACTI [5] to compute energy consumed in the L1 data/instruction caches. The experiments were conducted on Hopper, a Cray XE6 machine located at the National Energy Research Scientific Computing Center (NERSC). We used CrayPat (Cray's Performance Analysis Tool) [6] for instrumentation of application codes. We evaluated a simple matrix multiply code (PDGEMM) from the SCALAPACK library [7], a subset of the NAS Parallel Benchmarks (NPB) [8], three mini-apps from the Mantevo project [9, 10] namely - MiniFE, HPCCG and MiniGhost[11] , and a real-world HPC kernel GTC (Gyrokinetic Toroidal Code) [12] to measure dynamic and leakage energies in the L1 data/instruction caches. We also performed strong-scaling and weak-scaling experiments using HPCCG with two different data-set sizes to evaluate the impact of scaling on cache energy distribution. We also compared results in two different semiconductor process technology nodes to study the impact of how technology scaling affects cache leakage energies.

3. RESULTS

Based on the experiments, we observed that the leakage energy component of total cache energy varies from 40% to 80% for the L1 data cache and from 60% to 80% for the L1 instruction cache across a broad range of applications and other experiment factors. Our results indicate that leakage energy dominates dynamic energy in L1 caches, which are the smallest and most often accessed of all caches. With L2 and L3 caches being an order of magnitudes larger in size and less accessed than L1, their energy consumption is likely to be even more skewed towards leakage energy. Because most of the previously proposed cache leakage energy reduction techniques have an associated performance penalty limiting their use to higher levels of caches where penalties can be hidden in the cache latency, leakage energy in L1 caches is likely to remain a concern for system designers. The significance of these percentages emphasize the importance of addressing leakage as part of the exploration of architectural and circuit-level ideas to overcome the barriers to the next generation of extreme-scale computing. Our scheme for energy estimation in caches will help assess alternatives for such architecture exploration. Combining cache activity information obtained from performance tuning tools with an accurate cache energy modeling tool to generate fast and accurate energy consumption information for various levels of caches with minimal intervention and substantial lower overheads, the framework is easy to use.

Acknowledgements

This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. The authors thank the computational facilities provided by NERSC.

4. REFERENCES

[1] Titan: http://www.top500.org/system/177975.

[2] Peter Kogge et al. Exascale computing study: Technology challenges in achieving exascale systems. `http://users.ece.gatech.edu/mrichard/ExascaleComputingStudyReports/exascale_final_report_100208.pdf`, 2008.

[3] ITRS Roadmap. `http://www.itrs.net/`.

[4] S. Browne, J Dongarra, et al. A portable programming interface for performance evaluation on modern processors. *The Int. Journal of High Performance Computing Applications*, 14:189–204, 2000.

[5] Naveen Muralimanohar et al. CACTI 6.0: A Tool to Model Large Caches, Technical Report, HP Laboratories, April 2009. `http://www.hpl.hp.com/techreports/2009/HPL-2009-85.html`.

[6] Cray Performance Measurement and Analysis Tools. `http://docs.cray.com/`.

[7] L. S. Blackford et al. ScaLAPACK: A Portable Linear Algebra Library for Distributed Memory Computers - Design Issues and Performance. In *Supercomputing, 1996. Proceedings of the 1996 ACM/IEEE Conference on*, pages 5–5, 1996.

[8] D. H. Bailey et al. The NAS Parallel Benchmarks. Technical report, The International Journal of Supercomputer Applications, 1991.

[9] Michael A. Heroux et al. Improving performance via mini-applications. `http://www.sandia.gov/~maherou/docs/MantevoOverview.pdf`.

[10] Mantevo Project. `http://www.mantevo.org/`.

[11] Richard F. Barrett and Courtenay T. Vaughan, and Michael A. Heroux. MiniGhost: A Miniapp for Exploring Boundary Exchange Strategies Using Stencil Computations in Scientific Parallel Computing. `http://www.sandia.gov/~rfbarre/PAPERS/miniGhost.v1.0.pdf`.

[12] W. Lee. Gyrokinetic particle simulation model. *Journal of Computational Physics, 72(1):243-269*, 1987.