A Power API for the HPC Community

David DeBonis, Ryan E. Grant, Stephen L. Olivier, Michael Levenhagen,
Suzanne M. Kelly, Kevin T. Pedretti, and James H. Laros
Sandia National Laboratories*, Albuquerque, New Mexico
{ddeboni, regrant, slolivi, mjleven, smkelly, ktpedre, jhlaros}@sandia.gov

***Abstract*—Power measurement and control are necessary for the proper operation of future power capped HPC systems. Current approaches to power measurement and control are limited and have proprietary interfaces. This poster presents a new vendor-neutral API for power measurement and control, based on the API document developed at Sandia National Labs and reviewed by industry, laboratory, and university partners. The API provides a publicly available free interface design that can be used to leverage power measurement and control hardware for large systems. This poster presents a description of the API itself, and the motivation behind the design of the API and its individual interfaces. Finally, the poster summarizes several power related research projects at Sandia National Laboratories that can leverage the Power API to transition research efforts quickly and portably into a production domain.**

I. Introduction

Power and energy constraints are principal concerns for members of the HPC community – vendors, government laboratories, and universities. Their impact is expected to grow in extreme scale systems, where power caps are likely to be imposed. To mitigate the effects, different layers of the software stack, and different classes of users, will need to interact with vendor-supported power measurement and control capabilities on HPC platforms. However, there currently exists no common, portable API for such interactions. Our proposed API fills this gap, providing a vendor-neutral API for power measurement and control. A key point in our strategy has been community and vendor buy-in: Early reviewers of the API have included representatives from Intel, AMD, IBM, Cray, Adaptive, and other industry, laboratory, and university partners. This poster presents both an introduction to the origins and structure of the API itself, and also descriptions of our on-going R&D work in the power-aware computing domain that tie in to the API. While these are only a few examples, they point to the breadth and depth of innovation enabled by the API.

II. Overview of the API

When DOE/ NNSA deployed the Red Storm platform at Sandia in 2005, it ranked number 6 on the Top 500 list. It included foundational capabilities for power measurement and control that Sandia exploited to achieve power savings by profiling and then adjusting the power usage of production applications. For example, the SAGE application saw almost 50% power savings on a 4096-cores with only an 8% increase in run time by changing the p-state [1]. At the time, such mechanisms were rare and machine-specific. Now more vendors are including them in their hardware and system software, but using proprietary interfaces. Our prior experiences and vendor relationships (with Intel, AMD, IBM, etc.) have helped to inform our work on the vendor-neutral Power API.

Since a key goal of the Power API is to support all layers of the HPC software stack, we carried out an intensive use case study to consider the interactions between the system layers and between users and system layers. Informed by the use case study, the Power API defines a set of interfaces. Each interface expresses interactions between two system layers (e.g., operating system / monitor & control) or between a system layer and a person or entity (e.g., resource manager / user). The structure of each interface is the same, comprising the supported attributes and functions for that interface. This uniformity of design allows shared specification of shared core functionality, in addition to the individual specifications of functionality particular to each interface. It also enables the vendors working at multiple layers to maintain consistency in their implementations of the API.

The Power API is concerned with components of a platform with capabilites for power measurement and control, so a means of identifying system components is provided in the form of an object hierarchy. Example objects are platform, cabinet, node, and core, and the hierarchy need not be homogeneous. Object groups can be created, e.g., nodes in an allocation. Objects have associated attributes such as power and temperature, and statistics can be gathered for the attributes. Statistics have two modes, one for real-time measurement and another for gathering historical data from a data store. A metadata interface is provided that qunatifies properties of an object's attributes, e.g., how frequently internal sampling is performed for each measurement or how accurate the measurements are expected to be.

At over 120 pages in length, the full API is too long to completely describe in the poster (or this extended abstract). An example use case is shown on the poster, with accompanying source code. The example traverses the object hierarchy and reads and sets the power cap on a board.

III. Associated R&D Efforts

The poster highlights four representative R&D efforts taking place in conjunction with development of the Power API.

PowerInsight [2] is an out-of-band measurement device that is capable of collecting high-frequency current, voltage, power and energy measurements. PowerInsight was developed based

*Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

on prior experience in the area of large-scale power measurement for HPC and is an enabling technology for a variety of power related research. PowerInsight allows the exploration of different possible future energy measurement capabilities that may be offered in future commodity computing hardware.

Power and reliability trade-offs are first class design constraints for future large scale systems. The Power API facilitates monitoring and control of power consumption and this information can be applied to save power during resilience events [3] as well as informing and facilitating alternative resilience techniques [4].

The Power API is also facilitating new research on network power consumption. While networks typically have relatively flat power consumption, recent interest in methods of reducing network power consumption for next generation systems is ongoing. The Power API provides a common method of leveraging research in this area that can be easily applied to production systems that are Power API enabled.

Concurrency throttling research [5] for power savings as well as considerations of power and reliability in terms of energy saving techniques [6] have been enabled by real measurements of systems that can be enabled through the Power API. Such methods can also benefit from high-quality, high-frequency sampling and common methods of interfacing with measurement hardware.

IV. Related Work

Previous work have shown that thermal research on advanced aggresively-clock-gated super-scalar processors is problematic [7]. Component level profiling was achieved in another study utilizing ten multimeters per node, each monitoring power over a shunt resistor inserted through ATX extension cables [8]. While both results gathered accurate per node power data, the collection had to be made through individual collection over each node through multiple passes. Our instrumentation allows us to collect concurrently over all nodes regardless of scale in situ.

The implementation of a scalable power measurement framework was presented utilizing board level interface exploitations to measure power usage [9]. The concept of Application Power Signatures was introduced and the first quantitative study of OS noise was performed. Many of the concepts from that work were leveraged in our research studies.

Low-cost power monitoring devices that can operate inside commodity computing systems was introduced [10]. In their study it was noted that the use of AC monitors like PowerEgg and WattsUp were inadequate to capture fast variations in the DC load of the supply. The PowerInsight device was developed around the same timeframe but took a passive approach, eliminating CPU overhead costs and induced temperature rise.

Standard methods for unified access to hardware performance counters of microprocessors have been previously presented such as PAPI [11]. ACPI provides a standardized interface to Operating System-directed configuration and Power Management (OSPM) [12]. Both of these low-level methods have been shown to be useful and accepted. Such methods, though limited in scope, are not exclusive and can be easily adapted to the Power API abstract model.

V. Conclusion and Future Development

Partner organizations reviewed the API document in July 2014 and a larger audience provided feedback in September, following its official release [13]. Prototyping on test-bed platforms is an on-going effort. A production implementation of the API will be included in the deployment of the $174 million dollar ASC/NNSA Trinity platform. Moreover, we anticipate and encourage continuing community feedback to drive refinement of the API. It serves as a first step toward a vendor-neutral standardization of power measurement and control, which is sorely needed as the community grapples with the power and energy challenges of extreme scale HPC.

References

[1] J. H. Laros, III, K. T. Pedretti, S. M. Kelly, W. Shu, and C. T. Vaughan, "Energy based performance tuning for large scale high performance computing systems," in *Proceedings of the 2012 Symposium on High Performance Computing*, ser. HPC '12. San Diego, CA, USA: Society for Computer Simulation International, 2012, pp. 6:1–6:10.

[2] J. H. Laros, III, P. Pokorny, and D. DeBonis, "Powerinsight - a commodity power measurement capability," in *The Third International Workshop on Power Measurement and Profiling in conjunction with IEEE IGCC 2013*, Arlington Va, 2013.

[3] B. Mills, R. E. Grant, K. B. Ferreira, and R. Riesen, "Evaluating energy savings for checkpoint/restart," in *Proceedings of the 1st International Workshop on Energy Efficient Supercomputing*. ACM, 2013, p. 8.

[4] B. Mills, T. Znati, R. Melhem, K. B. Ferreira, and R. E. Grant, "Energy consumption of resilience mechanisms in large scale systems," in *2014 22nd Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP),*. IEEE, 2014, pp. 528–535.

[5] A. Porterfield, S. Olivier, S. Bhalachandra, and J. Prins, "Power measurement and concurrency throttling for energy reduction in OpenMP programs," in *IEEE 27th International Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW)*, 2013, pp. 884–891.

[6] R. E. Grant, S. L. Olivier, J. I. Laros, R. Brightwell, and A. K. Porterfield, "Metrics for evaluating energy saving techniques for resilient hpc systems," in *IEEE 28th International Parallel and Distributed Processing Symposium Workshops (HP-PAC), 2014*. IEEE, 2014, pp. 1–8.

[7] C. Isci and M. Martonosi, "Runtime power monitoring in high-end processors: Methodology and empirical data," in *Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO 36. Washington, DC, USA: IEEE Computer Society, 2003, pp. 93–. [Online]. Available: http://dl.acm.org/citation.cfm?id=956417.956567

[8] X. Feng, R. Ge, and K. Cameron, "Power and energy profiling of scientific applications on distributed systems," in *19th IEEE International Parallel and Distributed Processing Symposium, 2005. Proceedings*, 2005, pp. 34–34.

[9] J. Laros, K. T. Pedretti, S. M. Kelly, J. P. Vandyke, K. B. Ferreira, C. T. Vaughan, and M. Swan, "Topics on measuring real power usage on high performance computing platforms," in *IEEE International Conference on Cluster Computing and Workshops, 2009. CLUSTER'09.* IEEE, 2009, pp. 1–8.

[10] D. Bedard, M. Y. Lim, R. Fowler, and A. Porterfield, "PowerMon: Fine-grained and integrated power monitoring for commodity computer systems," in *IEEE SoutheastCon 2010 (SoutheastCon), Proceedings of the*, 2010, pp. 479–484.

[11] S. Browne, J. Dongarra, N. Garner, G. Ho, and P. Mucci, "A portable programming interface for performance evaluation on modern processors," *International Journal of High Performance Computing Applications*, vol. 14, no. 3, pp. 189–204, 2000.

[12] ACPI Specifications Team, "Advanced configuration and power interface specification," available at: http://acpi.info/spec.htm (Nov. 2013).

[13] J. H. Laros, D. DeBonis, R. Grant, S. M. Kelly, M. Levenhagen, S. Olivier, and K. Pedretti, "High Performance Computing - Power Application Programming Interface Specification, Version 1.0," Sandia National Laboratories, Tech. Rep. SAND2014-17061, August 2014.