HDEEM: High Definition Energy Efficiency Monitoring

Daniel Hackenberg, Thomas Ilsche,
Joseph Schuchart, Robert Schöne, Wolfgang E. Nagel
Technische Universität Dresden
Email: {daniel.hackenberg, thomas.ilsche, joseph.schuchart,
robert.schoene, wolfgang.nagel}@tu-dresden.de

Marc Simon, Yiannis Georgiou
Bull S.A.S.
Email: {marc.simon, yiannis.georgiou}@bull.net

***Abstract*—Accurate and fine-grained power measurements of computing systems are essential for energy-aware performance optimizations of HPC systems and applications. Although cluster wide instrumentation options are available, fine spatial granularity and temporal resolution are not supported by the system vendors and extra hardware is needed to capture the power consumption information. We introduce the High Definition Energy Efficiency Monitoring (HDEEM) infrastructure, a sophisticated approach towards systemwide and fine-grained power measurements that enable energy-aware performance optimizations of parallel codes. Our approach is targeted at instrumenting multiple HPC racks with power sensors that have a sampling rate of about 8 kSa/s as well as finer spatial granularity, e.g., for per-CPU measurements. We specifically focus on the correctness of power measurement samples and energy consumption calculations based on these power samples. We also discuss scalable and low-overhead or overhead-free options for online and offline (post-mortem) processing of power measurement data.**

I. Introduction

Energy efficiency of systems and applications has been an important research topic in recent years and its importance will continue to grow as the operating cost of large systems steadily increases. However, optimizing the properties of these systems is infeasible without in-depth knowledge about their behavior at run-time. A sophisticated measurement infrastructure is required to gain insight into energy and power characteristics. This includes both accurate and fine-grained measurements that help to find hot spots in parallel applications and to enable system optimizations at runtime to improve energy efficiency.

We define four criteria that describe the quality of a power measurement infrastructure. Fine *spatial granularity* is required to determine the components that consume the most energy, thereby revealing worthwhile targets for optimization. Fine *temporal granularity* is required for the analysis of short code regions and fluctuations in system behavior as they are crucial for the process of understanding these performance characteristics. High *scalability* is required for collecting system-wide information and enabling the analysis of highly parallel applications. High *accuracy* allows for reliable analyses, requiring both correctness and high measurement resolution. As a special aspect of *accuracy*, we define *energy correctness*. A series of power values for a certain time period is *energy-correct* if a correct energy value for the time period can be computed using these power values. Instantaneous power readings can be accurate on their own, but energy-incorrect as a series if they are not updated frequently enough. In such a case, additional power information between two updates would be required to compute the correct energy consumption.

Designing a measurement infrastructure that fulfills these requirements is a complex task. Previous power measurement approaches have often focused on power measurements for single nodes and small compound systems, with varying levels of detail. They typically use sensors that already exist in power supplies or on mainboards for power management purposes, e.g., power capping. We present challenges for accurate and fine-grained power and energy measurements before evaluating opportunities and limitations regarding cluster-wide per-node power measurements based on these integrated sensors. We also discuss our scalable data processing approach that is necessary for per-node power measurements with a high sampling rate when implemented at scale based on our current infrastructure and present our approach for its validation. We further provide an outlook on our planned setup that will provide researchers with accurate, scalable, and fine-grained measurements that help to analyze and optimize the energy efficiency of parallel applications.

II. Related Work

The four quality criteria for measuring power and energy are limited by two different factors. While *spatial granularity* is defined by the measurement implementation (physical location of the sensors), *scalability* is limited by the data processing infrastructure (e.g., how to store/process the amount of data). *Accuracy* and *temporal granularity* are limited by both, e.g., fine temporal granularity needs both capable sensors and scalable data processing.

Most new high-performance computing (HPC) systems implement ways to measure or estimate the energy consumption of components or compute nodes. Power supply units (*PSU*s) and power distribution units (*PDU*s) often provide energy measurements that can be accessed via the Intelligent Platform Management Interface (IPMI) [1]. These measurements are designed to monitor the hardware for administration purposes, energy-efficiency analysis is typically not a design target [2], [3]. Thus, the temporal resolution is usually in the range of 0.01 to 1 Sa/s [4], [5], [6], [7]. A recent accuracy analysis of two different PSU measurements is available in [8].

 This work has been published under DOI 10.1109/E2SC.2014.13, URL http://ieeexplore.ieee.org/document/7016382/

The same spatial granularity at node level is provided by *external measurements* using AC power meters. By using professional devices, e.g. from ZES Zimmer [9] or Yokogawa [10], high accuracy can be achieved [8]. They expose an update rate of up to 20 Sa/s to the user. The internal sampling rate is several orders of magnitude higher to capture the curve shape of current and voltage and to provide accurate averages for a given sampling interval instead of instantaneous measurements. Less professional power meters are usually not specified in such detail and may provide inaccurate data [11].

It is possible to measure *internal voltage lanes* in order to improve spatial granularity. PowerMon2 [12], [13], PowerPack [14], and PowerInsight [15] all use sensors to instrument these lanes and provide measurements with up to 1 kSa/s. PowerMon2 and PowerInsight specify the accuracy of their DC measurements with reference to the used sensors and the number of available bits. The currently specified accuracies are -6.6/+6.8 % and 1.8 %, respectively. They also specify update rates of 1024 Sa/s and "greater than 1 kHz (from user space)", respectively. To the best of our knowledge, detailed PowerPack specifications are not available.

Diouri et al. [11] and Hackenberg et al. [8] have analyzed the accuracy of internal measurements. However, internal instrumentation methods are intrusive and can possibly damage the hardware, making vendor support an important factor. IBM supports fine-grained spatial and temporal accuracy for power measurements on POWER and x86 servers using the Amester tool [7]. They use circuits that "place low impedance resistor in series with a power rail". The power information can be read with up to 1 kSa/s via the proprietary interfaces of Amester.

As part of their current XC30 system series, Cray offers an integrated power measurement infrastructure [16] that includes blade and GPU measurements at 10 Sa/s. The measurements are evaluated in detail in [17]. In addition to power and energy values, a *freshness counter* helps to determine whether the same value was read twice. However, the lack of information about when a value was read or updated makes it difficult to attribute readout values with correct time-stamps in order to correlate power usage with application activity. Since power readings are instantaneous values, aliasing is an issue if the application power has regular 100 ms patterns.

Some *devices* like current x86 processors and NVIDIA graphics cards [18] feature interfaces to read power and energy information. New values are reported with a rate of 100 Sa/s (AMD APM [19]), or 1 kSa/s (Intel RAPL [20], [21]). However, no timestamps are associated with power or energy samples. The effects of the missing timestamp information and the accuracy of the available information has been described in [8]. For processors that do not provide information on energy consumption, performance counter based models can be used to calculate estimates [22], [23], [24], [25]. Such measurements, however, have to be taken in-band and thus possibly interfere with the workload.

In addition to accurate measurement devices, a scalable software infrastructure is needed to store and process the power information. The Powerpack authors describe their software environment as scalable [14]. Its documentation does not mention full-system measurements but instead instructs users to select a single node for measurement. The Powerdam software is capable of processing power information from hundreds of nodes but with update rates of less than 1 Sa/s [26]. Laros et al. used PowerInsight on 104 nodes of a cluster [15], though with only one power sample per node and second.

While the PowerInsight approach is similar to ours, our implementation benefits from several different design decisions. On the hardware level, we use analog filters to overcome aliasing issues and noise. Additionally, we combine an FPGA and the existing BMC hardware instead of an autonomous measurement board. This reduces the hardware costs and complexity for the measurement equipment, removing an important obstacle on the path towards full-system measurement on large-scale systems. We also support the IPMI standard for gathering data at a low temporal granularity, allowing administrators to use their existing tools for gathering correct information. Our scalable measurement readout interface is based on energy values, which allows us to read values with different temporal granularity without losing information from missed samples. For fine-grained measurements, we provide the samples via the PCIe bus, which is faster then going over 10/100 MBit Ethernet or USB. Finally, we put much effort into calibrating our sensors, which is according to [15, Section IV.A] part of the future work on PowerInsight.

III. Power and Energy Measurement Challenges

Multiple information losses can occur when measuring power consumption. These deficits can originate from any point of the measurement chain, ranging from the current and voltage sensors, over the A/D conversion and potentially several steps of data processing, to the data storage. The correct interpretation of power measurement data can also be challenging and may require a detailed knowledge of the measurement approach and potential inaccuracies. Data interpretation becomes even more challenging in the presence of unknown or undocumented measurement errors.

Power measurements require current and voltage sensors that are typically used to capture instantaneous information at mostly regular intervals. Both sensor types exhibit a certain measurement error, e.g., due to manufacturing tolerances. The sensor signal will typically be subject to analog filtering to reduce aliasing and noise. Moreover, the signal needs to be sampled for conversion from the continuous time domain to discrete time steps. This is typically done using an analog-to-digital converter (ADC) that also performs the conversion from continuous-amplitude to discrete-amplitude signals. The power consumption of compute nodes (in particular CPUs) is highly dynamic due to the high operating frequencies, while the power measurement has far lower sampling rates. This means that a low-pass filter on the input side of the ADC is essential to satisfy the Nyquist-Shannon sampling criteria and to avoid aliasing effects. Otherwise the measured power samples cannot be used to compute energy consumption since they are not energy-correct. Detailed information regarding these filtering steps is usually not available for power measurement devices in HPC systems, making profound estimations of the final power measurement accuracy difficult.

The limited resolution of the ADC naturally results in a quantization error. Moreover, the conversion to machine processable data formats may introduce additional errors. The

IPMI standard is particularly prone to this type of error since it only specifies one byte of data information for a sensor reading [1, Chapter 35.14]. To overcome the limited range of an unsigned byte data type, a scaling factor and an offset can be defined [1, Chapter 36.3]. The introduction of a scaling factor significantly reduces the resolution of the power values. In our current installation, GPU nodes with high power requirements expose measurement data only in multiples of 7 W due to these limitations. To circumvent the reduced resolution, some vendors (e.g., Dell and Hewlett Packard [27], [28]) implement proprietary extensions to the IPMI protocol.

Without actually looking at energy measurements, we already identified a number of likely or even unavoidable sources for power measurement inaccuracies:

- voltage and current sensors
- lowpass filters (e.g., non-linearities)
- ADC quantization errors
- conversion to target data format for processing
- multiplication of voltage and current values within the limitations of the data format
- data processing, e.g., digital filters and calculation of averages

Each power consumption sample is typically associated with a timestamp. These timestamps are subject to the same datatype limitations as the power values. It is also often unclear how well the timestamp matches the exact time when the power sample was taken. Moreover, the sample may refer to an instantaneous power measurement or to an average over time that has been computed, e.g., by the Baseboard Management Controller (BMC). The resulting inaccuracies are depicted in Figure 1a and Figure 1b. This information – particularly internal sampling rates – is transparent to the user and most often undocumented. However, it is of utmost importance for assessing the correctness of the provided information.

Another important aspect is the calculation of energy-correct average power values over a certain time, and in turn energy consumption (product of average power and duration). Important aspects are

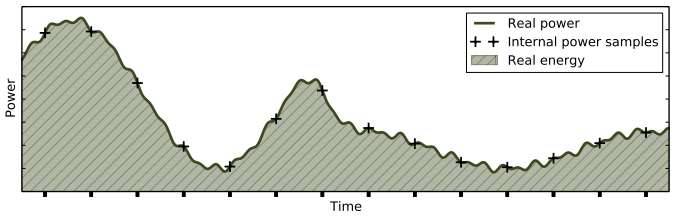
- How well does the timestamp match the actual time when the sample has been taken?
- Are the samples equidistant?
- Are the timestamps based on a different clock than the clock of the compute node? If yes, how well are these two clocks synchronized?

Furthermore, even though filters can improve the quality of the measurements, they often also introduce a certain delay. This means that the effect of load changes on a compute node will be visible in the power consumption profile later than they actually occurred. This delay has to be documented and it needs to be constant (i.e., not change over time) so that it can be accounted for during analysis.

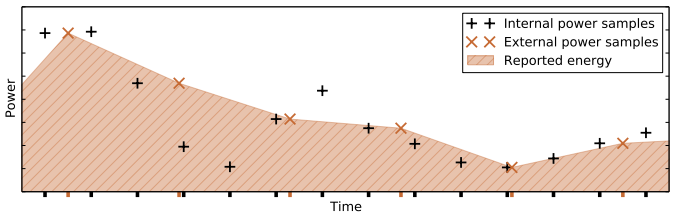
IV. Power and Energy Measurements on a SandyBridge Cluster in Production

We have a 270 node Intel Sandy Bridge based HPC cluster in production that served as test vehicle to evaluate options for scalable per-node power measurements using standard techniques. Each node provides nine different power sensors. Two power sensors are available per processor, one for the system agent (un-core) and one for core power consumption. Furthermore, four sensors measure the power consumption of the four DRAM channels and one sensor is attached to the 54 V DC power inlet to measure the overall node.

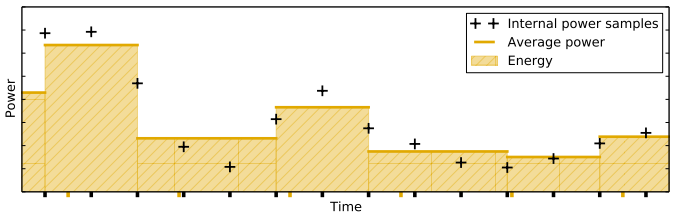
The values of the node sensor are passed through a low-pass filter (cutoff frequency of 0.16 Hz) before A/D conversion to prevent aliasing. The BMCs read the updated values from the sensors every ≈170 ms. So far, we have focused on the node sensors since the other sensors do not provide much value at this coarse temporal resolution. We query the values from the BMCs via IPMI at a rate of 1 Sa/s. All nodes are queried from only one management node via parallel FreeIPMI requests. Higher rates are possible, but they show only little dynamics due to the anti-aliasing filter and may cause performance issues due to IPMI-latencies and throughput limitations. However, for accurate energy consumption calculations, every single instantaneous value is required.



(a) internal sampling

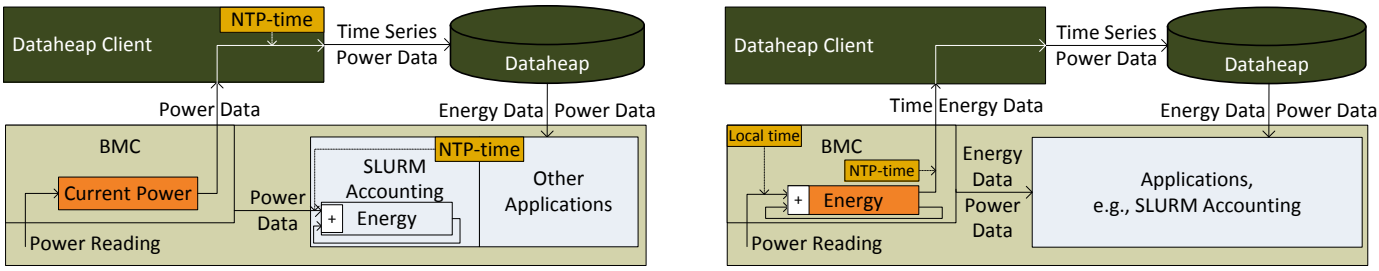


(b) External sampling with loss of timestamps and intermediate values in a typical IPMI scenario. Red ticks on the time axis show IPMI request times.



(c) Using the reported energy increase of the IPMI OEM extension to reconstruct average power. Yellow ticks on the time-axis show the IPMI request times, for which the last interval up to the most recent internal sampling timestamp is used.

Fig. 1: Information processing of six sampling steps, comparing internal sampling, external sampling using IPMI, and reconstruction of average power using the IPMI OEM extension.



(a) Standard implementation: missing measurement timestamps during data processing

(b) BMC extension: measurement timestamps used to provide an energy accumulation counter

Fig. 2: Integration of IPMI measurements in SLURM accounting and other applications

Some of the limitations described in Section III also apply: Even when reading the values via IPMI with 4 Sa/s or more, some information is missing since it cannot be determined when exactly the sensor readouts or updates occurred. It is therefore impossible to distinguish two successive sensor readings of identical value from no update. Furthermore, the granularity of the measurements via IPMI is 3 W for the node power consumption and 1 W for the other power sources due to the limitation of one byte for sensor readings in IPMI.

To overcome the IPMI limitations, we have devised an IPMI OEM extension and extended the BMC firmware to perform energy computations. With each internal measurement sample, the BMC will accumulate the energy consumption from the instantaneous power consumption readings using Equation 1.

$$E(t_0 \cdots t_n) = E_n = E_{n-1} + \frac{P(t_n) + P(t_{n-1})}{2}(t_n - t_{n-1}) \tag{1}$$

This IPMI extension allows us to atomically read the consumed energy E_n and the associated timestamp t_n with high precision. From two successive reads, an energy-correct average power over a well-defined time-period can be computed with Equation 2.

$$P_{avg}(t_{n-i} \cdots t_n) = \frac{E_n - E_{n-i}}{t_n - t_{n-i}} \tag{2}$$

The user can freely choose a measurement rate, even if intermediate internal samples are not read directly by the user (interval $i > 1$). This allows for a tradeoff between measurement overhead and fine temporal granularity, without sacrificing energy correctness. The resulting view on dynamic power consumption is shown in Figure 1c. Furthermore, it is very convenient and efficient to retrieve the energy consumed even over longer periods of time, e.g., the duration of a job, without intermediate queries. Measurements can be started and stopped using the IPMI extension. If used for multiple purposes, e.g., accounting and monitoring at the same time, the measurement should be running continuously. Statistics such as minimum/maximum instantaneous power are also available for each measurement. This information about power spikes can be used for worst-case power calculations. If the measurement runs for a long time, the energy values get very large in comparison to the small incremental changes. Due to the wide mantissa of double precision numbers, the introduced rounding error is negligible even for years of measurements. The granularity of values is only limited by the sensor itself.

In order to provide accurate temporal correlation, the BMC clocks and compute node clocks synchronize locally with the administrative nodes in regular intervals. Considering NTP accuracy in local networks and clock drift during the NTP refresh interval, the timestamps are accurate within 16 ms. This is sufficient for power readings every 170 ms. In contrast to the timestamps provided via the IPMI extension, the energy calculation is done using a local non-NTP monotonous time to avoid anomalous energy values in case of non-monotonous timestamps due to larger NTP corrections. The effect of using different clocks for computing average power is negligible during normal operation, as the involved quartz are specified for errors below 0.001 %. Figure 2 compares the standard measurement implementation and the new BMC extension.

In summary, the main improvements of the BMC and IPMI extension are threefold. First, the systematic 4 Sa/s internal sampling is significantly higher than the frequency of the low-pass filter and thus all power variation is integrated to provide correct energy and average power values with negligible aliasing. Second, using a protocol extension, the data format does not limit the accuracy. Third, accurate timestamps are provided with the values via atomic reads to allow for correct correlation of power readings with application phases or system events as well as correct energy computations.

V. Integration into HPC Environment

Users can access the energy readings from the BMCs through two different interfaces: (I) the resource and job management system SLURM [29] and (II) the distributed data collection and monitoring tool Dataheap [30].

A. Resource and Job Management System Integration

The Simple Linux Utility for Resource Management (SLURM) is an open-source resource and job management system specifically designed for the scalability requirements of

Listing 1: Sample output of the sacct utility showing the energy consumed by a job running for 4 hours.

JobID	NTasks	Start	End	ConsumedEnergy
5876326		2014-04-02T08:40:20	2014-04-02T12:40:45	
5876326.bat+	1	2014-04-02T08:40:20	2014-04-02T12:40:47	21
5876326.0	2048	2014-04-02T08:40:21	2014-04-02T12:40:50	610.66M

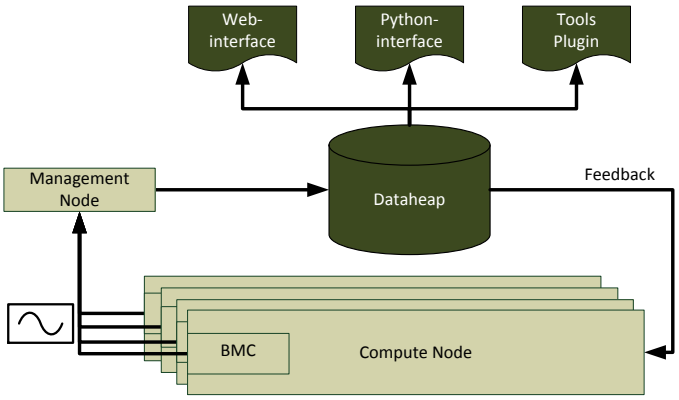


Fig. 3: Dataheap and measurement infrastructure.

state-of-the-art supercomputers. SLURM provides functionalities that enable power monitoring per node as well as power and energy accounting per job based on IPMI and RAPL measurement interfaces [31]. These functionalities have been extended to support the new BMC firmware for HDEEM using the FreeIPMI library. The SLURM accounting utility `sacct` provides users with an easy way to determine the energy consumption of each job step. Listing 1 shows sample output of this utility for a job with four hours of runtime and 610.66 MJ energy consumption. By providing information about the energy required to run jobs, users can estimate the impact of code optimizations with respect to energy consumption.

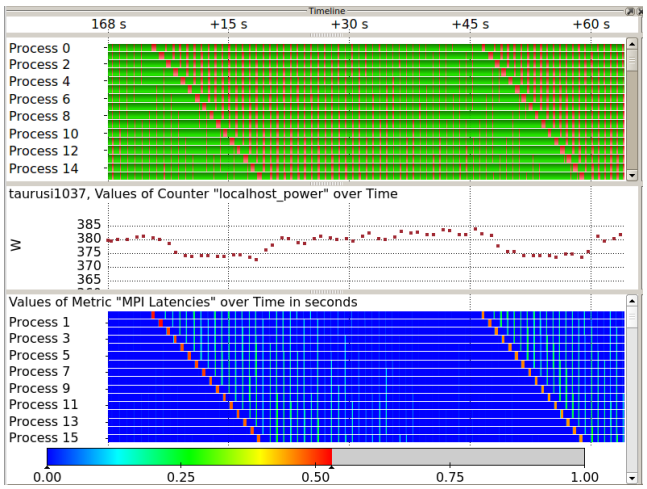
More detailed information can be gathered using the SLURM profiling option `--profile`. Through a post-mortem analysis of the generated file in HDF5 format, users can get profiling information of their power usage with a sampling frequency of 1 Sa/s. Other SLURM plugins feed additional information into the same HDF5 file, e.g., about the Lustre file-system and the network communication.

B. Data Collection and Monitoring Tool Integration

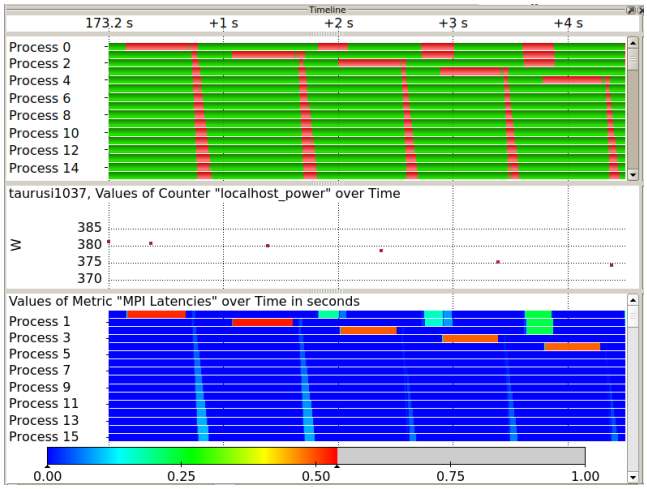
The Dataheap infrastructure was designed as a scalable infrastructure for storing and processing continuous measurement data. It is therefore suitable as a storage infrastructure for the power measurements that are online regardless of running jobs. This allows users and administrators to access the data for online as well as post-mortem analysis and does not require manual intervention to enable the recording.

The integration of the Dataheap infrastructure is depicted in Figure 3. The power measurement data is collected from the BMCs and pushed to the Dataheap by a management node. From there, the data is accessible through a number of different interfaces. Users can browse through the data using a web interface that allows them to set arbitrary timeframes and to combine the data from multiple measurement sources in one display. This way, the system behavior can be analyzed and compared over a long time period and the effects of changes to the system can be tracked based on historical data.

The data can also be accessed through an API that enables the use of automatic data evaluation tools. At the moment, the interface is available for manual analysis via Python scripts and as plugins for the performance measurement infrastructures VampirTrace [32] and Score-P [33]. The plugins integrate the power measurement data into application performance traces and provides an easy way of correlating the behavior of a parallel application with the power and energy measurements. Figure 4a depicts a trace with integrated power measurements, which shows repeating patterns of a Linpack run with power consumption drops when all processes running on that node exhibit long MPI wait states. After these long wait phases, communication is still performed but the power consumption of the node increases again.



(a) 64 s segment with a pattern correlating long MPI wait states with a drop in power consumption of the node.



(b) 4 s segment demonstrating the lack of temporal resolution for fine-grained analysis.

Fig. 4: Vampir visualizations of a trace of a Linpack run containing the application behavior (top, including MPI communication in red and computation in green), power consumption of the node that ran processes 0-15, and a heatmap of the MPI wait states.

VI. Verification of Measurements

For our HPC cluster, we have performed an in-depth verification of the energy measurement and energy accounting capabilities. The setup for the verification of one chassis is depicted in Figure 5. Reference measurements were conducted using two calibrated ZES Zimmer LMG450 power analyzers (four channels each). They were attached to the four 54 V inputs (two channels per input) that provide power to the entire chassis. In this configuration, the ZES power meters are specified with a maximum relative error of 0.5 % of the measured value plus 0.5 % of the measuring range. Since all inputs form one internally connected lane, only the total sum of power over all measurement channels was of interest. Each of the eight measurement channels is limited to a maximum current of 16 A, resulting in a theoretical limit of 6.9 kW for all eight channels. The practical limit was lower due to imbalanced channel loads, resulting in the use of only 12 out of 18 available nodes in the chassis.

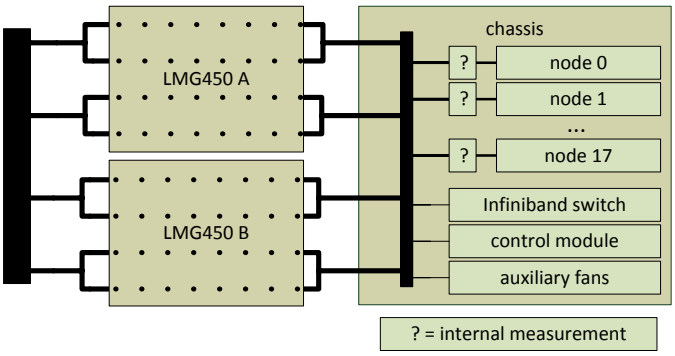


Fig. 5: Measurement setup to verify vendor measurements with precision power meters in the 54V DC power feed.

The main goal of the test was to verify the SLURM energy accounting values using the reference measurements. In addition, we also compared them with the Dataheap recordings. The test workloads for the compute nodes include:

- 5 minutes of idle (sleep, single nodes),
- 5 minutes of high load (Firestarter [34], single nodes),
- a regularly alternating synthetic high/low load code with specific interval lengths to trigger aliasing,
- Linpack (single nodes, 12 nodes), and
- a user application (12 nodes).

The initial verification run revealed significant aliasing issues within the SLURM accounting. Errors of up to 49 % occurred when we deliberately created worst case scenarios using our high/low load benchmark. The aliasing was caused by the IPMI under-sampling effects described in Section III. In this setup, the SLURM plugin used instantaneous power samples every 3 s to compute energy. We therefore developed the IPMI extension described in Section IV and integrated it into SLURM. Furthermore, a few nodes exhibited strong calibration issues of up to 15 W [1] (28 % for an idle node). This was fixed by a cluster-wide calibration.

[1] Since energy values are compared, the actual absolute errors are in Joules. We normalize those to average power in Watts.

In the final verification of the improved measurement and integration, five nodes were studied in detail with only one running at a time. The largest error found was 3.7 W (6.3 %) for idle nodes and 9.9 W (2.6 %) under full load. While running Linpack and the user application in parallel on 12 active nodes in the chassis, the overall error was consistently below 0.5 %. Neither Linpack nor the user application measurement suffered from aliasing effects and the remaining calibration issues leveled each other out across all nodes. There was no significant difference between the energy values reported by SLURM and those computed from Dataheap recordings. Table I lists the errors of all tested nodes.

Target	Idle	Firestarter	Linpack
node 1	-2.9 %	-0.4 %	-0.6 %
node 2	-2.4 %	0.4 %	0.2 %
node 3	-6.3 %	-1.3 %	-1.2 %
node 4	-3.3 %	0.4 %	0.2 %
node 5	-1.1 %	2.6 %	2.3 %
12 nodes	n/a	n/a	-0.2 %

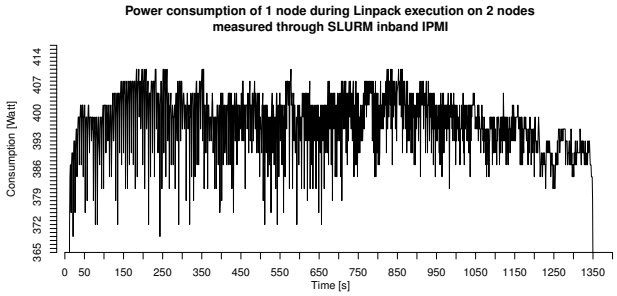
TABLE I: Relative deviation of SLURM energy accounting compared with reference measurement. For each entry, the largest absolute error of all repeated runs is shown.

The default SLURM plugin for calculating the job energy consumption, `acct_gather_energy/ipmi`, collects instantaneous power consumption values per node through the FreeIPMI *libipmimonitoring* API [31]. The problem of that plugin is that it cannot collect more than 1 Sa/s. Hence, in order to consider the new IPMI extensions developed for the firmware of the BMC (see Section IV), a new SLURM plugin was implemented based on the FreeIPMI *ipmi_raw* API. The new `acct_gather_energy/ipmi_raw` plugin enables the collection of the energy consumption values directly from the BMC, including the advantage of the internal polling rate of 4 Sa/s. We have performed experiments to compare the accuracy and overhead between the different SLURM monitoring modes. The experiments have been executed using two nodes with Intel Ivy Bridge processors (dual socket, 12 cores per socket, 64 GB memory per node).

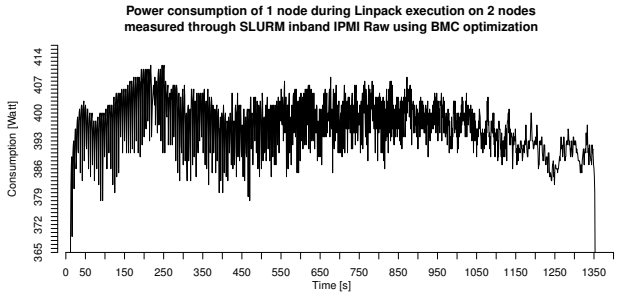
Listing 2 shows the energy consumed from Linpack executions using the two different energy accounting plugins. While the reported overall energy is similar, Figure 6 shows that the new ipmi_raw plugin uses more accurate measurement values. Those figures have been plotted with the measurements collected through the profiling utility of SLURM with 1 Sa/s. Figure 6a reflects the power consumption of the job `xhpl_ipmi` that uses the default energy accounting plugin while Figure 6b depicts the power consumption of job `xhpl_ipmi_raw` that makes use of the optimized plugin. In the optimized energy accounting plugin, the power measurements are more accurate.

Listing 2: SLURM `sacct` utility showing the energy consumed by Linpack executions, measured through two different monitoring modes

JobName	State	Elapsed	ConsumedEnergyRaw
xhpl_ipmi	COMPLETED	00:22:39	1070894.000000
xhpl_ipmi_raw	COMPLETED	00:22:42	1073068.000000



(a) default ipmi plugin, resolution of 3 W, instantaneous values



(b) new ipmi_raw plugin, floating-point resolution, averaged values

Fig. 6: Visualization of SLURM Power Profiling of one node during a Linpack execution on 2 nodes

They reflect the average power consumption of at least 4 samples and are not limited by the IPMI 3 W granularity. This explains the higher sensitivity of power values observed in Figure 6b.

Figure 7 shows the overhead of the different monitoring modes on the first compute node during the Linpack executions. A reference measurement without accounting/profiling is compared to both plugins (*ipmi* and *ipmi_raw*), each with accounting only and with accounting+profiling enabled. Linpack performance and execution times remain stable in all cases. The CPU-time is significantly reduced with *ipmi_raw*. This is a result of the BMC-internal polling in contrast to the default *ipmi* case where a SLURM thread is responsible for the polling. The memory usage is only slightly reduced using the *ipmi_raw* plugin, which is expected since the profiling mode always requires memory for logging the power information for the node and for generating the hdf5 files. In summary, the overhead of the improved *ipmi_raw* plugin is negligible when used with accounting only. Profiling provides additional information at the cost of some performance (1s runtime overhead for a >20 min application runtime using the improved plugin). The measurement data processing is decentralized on the compute nodes, therefore no scalability bottleneck is introduced. We use *ipmi_raw* energy accounting on our full 270 nodes production system.

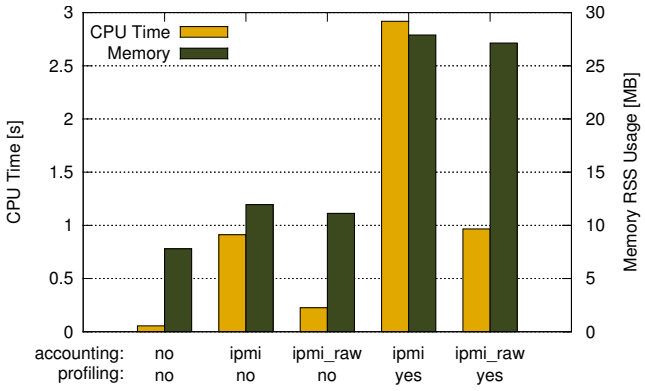


Fig. 7: CPU and memory overhead for different SLURM monitoring modes during a job with ≈1360 s duration.

VII. Increasing Granularity and Accuracy of Power and Energy Measurements

For our two current installations – an AMD Opteron cluster with 92 nodes and an Intel Sandy Bridge cluster with 270 nodes – we measure and store the power consumption at node level with a sampling rate of 1 Sa/s. We extended our performance analysis tools to be able to relate the power information to application traces. As we show in Figure 4a, this information can provide some insight, but a higher temporal granularity will be beneficial for understanding and optimizing power relevant application behavior. This shortcoming of the current solution is demonstrated in Figure 4b, where the temporal resolution of the power measurements is too low to gain detailed insights for a 4 s execution phase. This makes it impossible to distinguish power measurements for different code regions of the application, e.g., computational functions and communication. Moreover, a higher spatial granularity would allow users to identify and target specific components for optimization. Separating memory and processor power, for example, enables users to target the processor for energy tuning, which is the only component that is accessible for optimizations from a software perspective. Nevertheless, gathering information about the memory energy consumption can provide insights into the impact of certain hardware settings on the memory, e.g., changing the CPU and memory frequencies. It also enables administrators to find power anomalies where similar components differ in their power dissipation.

For our next (Intel Haswell based) installation we will improve temporal and spatial granularity as well as accuracy significantly. Each of the nodes will provide seven power sensors that measure

- the total power consumption of the node,
- the two voltage lanes of the processors (V_{CCIN}), and
- the four lanes of the memory DIMMs.

Figure 8 shows the position of sensors in the node architecture. The node power consumption is measured at the 54 V DC input with a rate of 8 kSa/s. Processor and DIMM power are measured at the level of the voltage regulators (VR) with a rate of 1 kSa/s each. The obtained samples are smoothened using a digital finite impulse response (FIR) filter. This digital filter reduces the number of samples that need to be processed

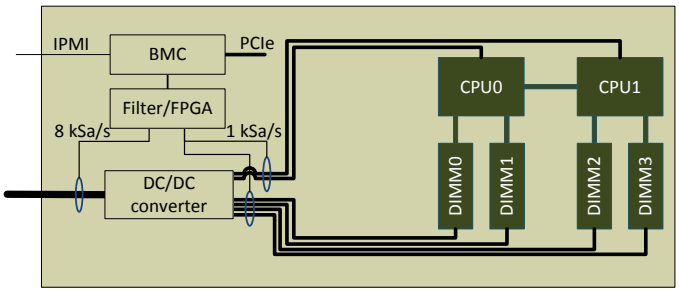


Fig. 8: Measurement points within one compute node with the improved architecture.

without compromising energy-correctness and the representation of dynamic behavior. It computes the weighted average of the incoming samples, putting the highest weight on the values in the middle of the pipe. This causes sharp variations of the input signal to become visible in the output with a certain delay that depends on the FIR's pipe length and the frequency of the incoming signal. As the raw data sampling frequency and the filter pipe length differ (70 elements for voltage lanes, 90 elements for power input), two different but constant delays are introduced that can be coped with in software during analysis.

The filtered values will be accessible for external software through several interfaces. Some measurement data will still be available through IPMI, including live power consumption values and energy consumption through the extended OEM interface, as described in Section IV. More detailed data will be available via PCIe and a C library. This includes statistical values containing minimum, average, and maximum power consumption for each power source as well as time lines with timestamped filtered samples for detailed analysis, e.g., for integration into performance analysis tools and the Dataheap infrastructure.

Besides the improved spatial and temporal resolution, we also focused on achieving a high relative accuracy. Due to the large range of measurement values from low to high load, a high accuracy is hard to achieve. On the one hand, we targeted a 2 % for the blade sensor, which was first obtained only for measurement values larger than 300 W. Only by applying a linear calibration to the measured value, we were able to improve the accuracy to 1 % for all values above 50 W, which includes the consumption of an active idle blade (powered on, no load). The coefficients for this correction have to be determined for each blade as they depend on the individual sensor. On the other hand, the VR sensors have a lower accuracy and their response curve is not linear. For the VR sensor measuring the processor's input power, the error can be as much as 15 % for low values. By applying a linear correction, we can achieve an accuracy of 3 % in the range of 20 W to 175 W, which matches the power consumption of a processor being idle or under load, respectively.

While the final measurement solution is not yet available, a prototype version has already been deployed and evaluated. The prototype differs from the final product in the following ways: (I) Start and stop signals are sent through the BMC, using IPMI commands. In the final version, the data will be accessible via PCIe. (II) The fine-grained measurement data is stored in FPGA memory. Due to limited memory, the prototype is only capable of holding data for a 16 s time window. In the future, the data will be buffered in the BMC, which provides enough space for data of eight hours. (III) The FPGA has exclusive access to the power sources, thus prohibiting parallel access to the sensor data from the BMC during FPGA measurement. In the final version, both the BMC and the FPGA will be able to access that data concurrently. (IV) The blade sensor signal is filtered by an analog low-pass, first-order filter with a cutoff frequency of 400 Hz (20 dB/decade). The final version will use a second-order filter at 500 Hz. (V) The A/D converters in the final version will have a more fine-grained resolution of 0.22 W.

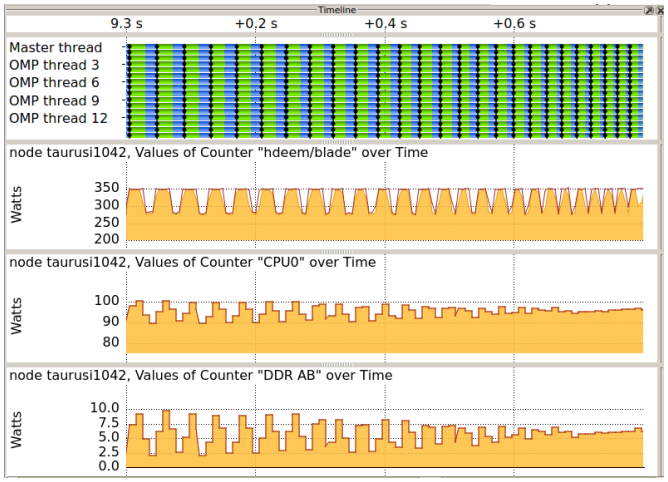
To demonstrate the capabilities of the prototype, we have created a high/low alternating workload with decreasing period lengths using an artificial benchmark. The experiments were conducted on a node equipped with two eight-core Intel Sandy Bridge E5-2690 CPUs clocked at 2.90 GHz and 32 GB RAM distributed over 4 DDR channels. Figure 9a shows the results of this benchmark with period lengths ranging from 60 ms down to 18 ms, including the overall blade consumption at 1 kSa/s together with one of the two CPU sockets and one DRAM channel at 100 Sa/s each. The data demonstrates the different temporal resolution of the measurements of the blade and the VRs (CPU sockets and DRAM). The alternating load is visible in the VR measurements up to a minimum period length of approx. 30 ms (low and high load each 15 ms), which is about three times the temporal resolution of the filtered VR measurement data. For the blade measurement with 1 kSa/s, a clear distinction between high and low load is possible for period lengths as low as 5 ms (not depicted).

A measurement of a real-world application is shown in Figure 9b with a trace of the NAS Parallel Benchmark BT-MZ containing power measurement data from the prototype, again for the blade, one CPU, and one DRAM channel. Both the VR and the blade measurements reflect the lock-step properties of the application execution, marking the communication phase with a drop in power consumption due to MPI wait states. However, this has only been a short class B run and it remains as future work to measure longer application runs.

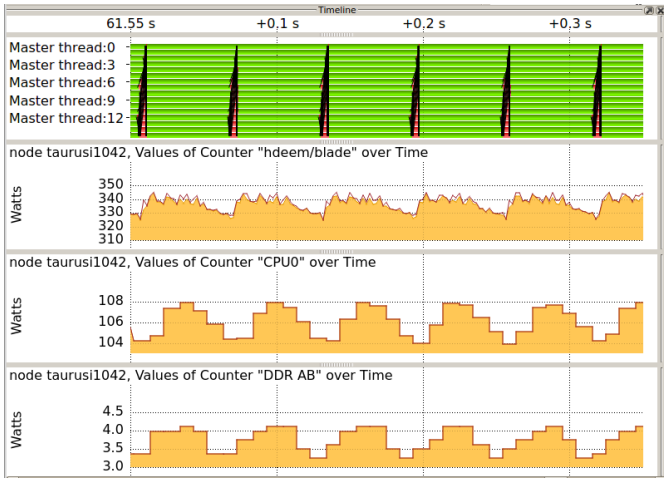
Both examples demonstrate the capabilities of our prototype to deliver fine-grained measurements for a detailed analysis of an application's behavior with respect to power and energy consumption. This now allows us to isolate the energy consumption of shorter application phases and parallel functions with runtimes in the order of a few millisecond, e.g., to optimize the behavior of MPI wait states, thread synchronization primitives, or certain computational kernels to be more energy-efficient.

VIII. Conclusion and Future Work

In this paper, we discuss the state-of-the-art of power and energy consumption measurement infrastructures for HPC systems. For this evaluation, we define four quality criteria: spatial granularity, temporal granularity, accuracy, and scalability. We describe common issues regarding these criteria that our existing power and energy measurement infrastructures and all similar projects are facing. Furthermore, we present first results of our approach to tackle these issues within the



(a) Synthetic benchmark with alternating high/low load periods ranging from 43 ms to 18 ms; high load phases colored green, low load phases in blue.



(b) NPB BT-MZ class B; computation phases in green, MPI functions in red, messages as black lines.

Fig. 9: Visualization of traces containing energy measurements from the the HDEEM prototype, including the blade power consumption, one CPU socket, and one DRAM channel.

HDEEM project. While spatial and temporal granularity are defined mostly by the hardware implementation, many of the accuracy issues that can be found upon closer evaluation stem from drawbacks in the data processing software that can be dealt with. In our existing 270-node cluster, we were able to reduce the measurement error for multi-node real-world application runs to less than 0.5 %.

Our plans for an improved measurement infrastructure include a dedicated measurement FPGA that will be installed on every blade. With this solution, we will be able to improve spatial granularity to measure blade, CPU, and DRAM power consumption separately. We will improve temporal granularity to up to 1 kSa/s, and plan to showcase scalability up to more than 500 nodes. Moreover, our accuracy target is 2 % and explicitly includes not only power consumption but also energy consumption, which requires a well-designed filtering approach to deal with aliasing effects.

The evaluation of an early prototype solution shows promising results in terms of temporal and spatial granularity. While the hardware development is mostly finished with small exceptions such as filter design, our future work will focus on software advancements that should enable scalability for cluster-wide measurements with negligible overhead due to out-of-band data transfer and processing.

Acknowledgment

Parts of this work have been supported by the Deutsche Forschungsgemeinschaft (DFG) via the Collaborative Research Center 912 “Highly Adaptive Energy-Efficient Computing” (HAEC, SFB 921/1 2011). The authors would like to thank Mario Biehlert for his contributions.

References

[1] “IPMI Specification, V2.0, Rev. 1.1,” http://www.intel.com/content/www/us/en/servers/ipmi/ipmi-second-gen-interface-spec-v2-rev1-1.html, Intel, Hewlett-Packard, NEC, Dell, Tech. Rep., 2013.

[2] “Oracle Integrated Lights Out Manager (ILOM) 3.0 Daily Management - Concepts Guide,” http://docs.oracle.com/cd/E19860-01/.

[3] “IBM Systems Director Active Energy Manager Installation and User’s Guide,” http://www-03.ibm.com/systems/management/director/about/director52/about52/resources/.

[4] “New IBM switched and monitored family of power distribution units makes it easy to protect and manage high-availability rack-based systems,” IBM Corporation, Tech. Rep., 2010. [Online]: http://www-01.ibm.com/common/ssi/rep_ca/1/899/ENUSLG10-0081/ENUSLG10-0081.PDF

[5] *PowerEdge R720 and R720xd Technical Guide*, Dell Inc, 2012.

[6] *ClustSafe - Energy Management and Current Distribution with Full Control (Datasheet)*, MEGWARE Computer Vertrieb and Service GmbH, 2012.

[7] L. Brochard, R. Panda, D. DeSota, F. Thomas, and R. H. Bell, Jr., “Power and energy-aware processor scheduling,” in *Proceedings of the 2Nd ACM/SPEC International Conference on Performance Engineering (ICPE)*, 2011. [Online]: http://dx.doi.org/10.1145/1958746.1958780

[8] D. Hackenberg, T. Ilsche, R. Schöne, D. Molka, M. Schmidt, and W. E. Nagel, “Power measurement techniques on standard compute nodes: A quantitative comparison,” in *Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on*, 2013. [Online]: http://dx.doi.org/10.1109/ISPASS.2013.6557170

[9] *4 Channel Power Meter LMG450, User manual*, ZES ZIMMER Electronic Systems, 2009.

[10] *WT500 Power Analyzer User’s Manual*, http://www.yokogawa.co.jp/ftp/dist/ks/im/en/bmi/IM760201-01E.pdf, Yokogawa Meters and Instruments Corporation.

[11] M. Diouri, M. Dolz, O. Glück, L. Lefvre, P. Alonso, S. Cataln, R. Mayo, and E. Quintana-Ort, “Solving some mysteries in power monitoring of servers: Take care of your wattmeters!” in *Energy Efficiency in Large Scale Distributed Systems*, ser. Lecture Notes in Computer Science, 2013. [Online]: http://dx.doi.org/10.1007/978-3-642-40517-4_1

[12] 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. [Online]: http://dx.doi.org/10.1109/SECON.2010.5453824

[13] D. Bedard, M. Y. Lim, R. Fowler, and A. Porterfield, “Powermon 2: Fine-grained, integrated power measurement,” Renaissance Computing Institute (RENCI), Tech. Rep. TR-09-04, 2009.

[14] R. Ge, X. Feng, S. Song, H.-C. Chang, D. Li, and K. W. Cameron, “Powerpack: Energy profiling and analysis of high-

performance systems and applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 5, 2010. [Online]: http://dx.doi.org/10.1109/TPDS.2009.76

[15] J. H. Laros III, P. Pokorny, and D. DeBonis, "Powerinsight-a commodity power measurement capability," May 2013. [Online]: http://www.sandia.gov/~jhlaros/Laros_23_PowerInsight.pdf

[16] S. Martin and M. Kappel, "Cray XC30 Power Monitoring and Management," *Proceedings of CUG2014*, 2014.

[17] A. Hart, H. Richardson, J. Doleschal, T. Ilsche, M. Bielert, and M. Kappel, "User-level Power Monitoring and Application Performance on Cray XC30 supercomputers," *Proceedings of CUG2014*, 2014.

[18] V. M. Weaver, M. Johnson, K. Kasichayanula, J. Ralph, P. Luszczek, D. Terpstra, and S. Moore, "Measuring energy and power with papi." in *ICPP Workshops*, 2012. [Online]: http://dx.doi.org/10.1109/ICPPW.2012.39

[19] R. Jotwani, S. Sundaram, S. Kosonocky, A. Schaefer, V. Andrade, G. Constant, A. Novak, and S. Naffziger, "An x86-64 core implemented in 32nm soi cmos," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 2010. [Online]: http://dx.doi.org/10.1109/ISSCC.2010.5434076

[20] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanaa, and C. Le, "RAPL: memory power estimation and capping," in *International Symposium on Low Power Electronics and Design (ISLPED)*, 2010. [Online]: http://dx.doi.org/10.1145/1840845.1840883

[21] E. Rotem, A. Naveh, D. Rajwan, A. Ananthakrishnan, and E. Weissmann, "Power-management architecture of the intel microarchitecture code-named sandy bridge," *Micro, IEEE*, vol. 32, no. 2, 2012. [Online]: http://dx.doi.org/10.1109/MM.2012.12

[22] G. Contreras and M. Martonosi, "Power prediction for intel xscale reg; processors using performance monitoring unit events," in *International Symposium on Low Power Electronics and Design (ISLPED)*, 2005. [Online]: http://dx.doi.org/10.1109/LPE.2005.195518

[23] R. Joseph and M. Martonosi, "Run-time power estimation in high performance microprocessors," in *International Symposium on Low Power Electronics and Design (ISLPED)*, 2001. [Online]: http://dx.doi.org/10.1145/383082.383119

[24] K. Singh, M. Bhadauria, and S. A. McKee, "Real time power estimation and thread scheduling via performance counters," *SIGARCH Comput. Archit. News*, vol. 37, no. 2, 2009. [Online]: http://dx.doi.org/10.1145/1577129.1577137

[25] B. Goel, S. McKee, R. Gioiosa, K. Singh, M. Bhadauria, and M. Cesati, "Portable, scalable, per-core power estimation for intelligent resource management," in *IGCC*, 2010. [Online]: http://dx.doi.org/10.1109/GREENCOMP.2010.5598313

[26] H. Shoukourian, T. Wilde, A. Auweter, and A. Bode, "Monitoring power data: A first step towards a unified energy efficiency evaluation toolset for HPC data centers," *Environmental Modelling & Software*, no. 0, 2013. [Online]: http://dx.doi.org/10.1016/j.envsoft.2013.11.011

[27] *Using the BMC Management Utility*, Dell Inc. [Online]: http://support.dell.com/support/systemsinfo/document.aspx?s=slg&file=/software/smbmcmu/1.2/en/ug/bmcugc0d.htm

[28] *IMM and IMM2 Support on IBM System x and BladeCenter Servers*, 2014. [Online]: http://www.redbooks.ibm.com/abstracts/tips0849.html?Open

[29] A. B. Yoo, M. A. Jette, and M. Grondona, "SLURM: Simple Linux utility for resource management," in *Job Scheduling Strategies for Parallel Processing*, 2003. [Online]: http://dx.doi.org/10.1007/10968987_3

[30] M. Kluge, D. Hackenberg, and W. E. Nagel, "Collecting distributed performance data with dataheap: Generating and exploiting a holistic system view," *Procedia Computer Science*, vol. 9, no. 0, 2012. [Online]: http://dx.doi.org/10.1016/j.procs.2012.04.215

[31] Y. Georgiou, T. Cadeau, D. Glesser, D. Auble, M. Jette, and M. Hautreux, "Energy accounting and control with slurm resource and job management system," in *ICDCN*, 2014. [Online]: http://dx.doi.org/10.1007/978-3-642-45249-9_7

[32] R. Schöne, R. Tschüter, T. Ilsche, and D. Hackenberg, "The VampirTrace plugin counter interface: introduction and examples," in *Euro-Par Parallel Processing Workshops*, 2010. [Online]: http://dx.doi.org/10.1007/978-3-642-21878-1_62

[33] A. Knüpfer, C. Rössel, D. Mey, S. Biersdorff, K. Diethelm, D. Eschweiler, M. Geimer, M. Gerndt, D. Lorenz, A. Malony, W. E. Nagel, Y. Oleynik, P. Philippen, P. Saviankou, D. Schmidl, S. Shende, R. Tschüter, M. Wagner, B. Wesarg, and F. Wolf, "Score-p: A joint performance measurement run-time infrastructure for periscope, scalasca, tau, and vampir," in *Tools for High Performance Computing 2011*, 2012. [Online]: http://dx.doi.org/10.1007/978-3-642-31476-6_7

[34] D. Hackenberg, R. Oldenburg, D. Molka, and R. Schöne, "Introducing FIRESTARTER: A processor stress test utility," in *International Green Computing Conference (IGCC)*, 2013. [Online]: http://dx.doi.org/10.1109/IGCC.2013.6604507