D3. Scope and Research Question

Dissecting the software-based measurement of CPU energy consumption

David Botton Léandre Gerday Ferenc Varga

Group Linus Torvalds, INFO B302 - IDS (2024 - 2025)

1 Scope and Objectives

1.1 Scope

This paper aims at comparing and understanding available means in the matter of `measuring CPU consumption in energy-sensitive computing domains`, based on the Zettelkasten method[1].

The Zettelkasten approach structures our analysis through three core principles:

- **Literature Notes** capturing atomic insights from source material (e.g.: RAPL's overflow correction in [3, §III-B]).
- **Topic Notes** synthesizing cross-cutting concepts such as "Energy Measurement Overhead".[2]
- **Bidirectional Connections** between implementation challenges (e.g.: timing inaccuracies) and architectural trade-offs.

This document treats primarily of Intel's Running Average Power Limit (RAPL) technology, a critical tool for measuring high-performance devices such as distributed systems. RAPL is organized around four foundational mechanisms; direct Model-Specific Register (MSR) interaction, Linux Power Capping Framework (powercap), Performance Counters (perf-events), and extended Berkeley Packet Filter (eBPF). Together, they allow us to evaluate technical trade-offs, any performance overhead, and to measure possible challenges specific to an implementation. This document focuses on `software-driven approach` rather than hardware-based methods (e.g.: power meters) or statistical estimation models, as the former present much greater reproducibility, thus allowing more comprehensive testing benchmarks, and the provisioning of a larger variety of documentation. In that matter, RAPL provides direct, hardware-integrated energy counters without requiring external sensors.[3] By narrowing the focus to RAPL-based methods, is addressed the software-based CPU consumption measurement, particularly under high-frequency sampling and parallelized workload.[4]

Beyond RAPL, we also consider complementary software-based power meters [5]. These include PowerAPI, Scaphandre, and Energy Scope. Each offers different energy measurement granularities. Some provide process-level data. Others focus on system-wide profiling. GPU energy measurement tools are also relevant for complete system analysis.

1.2 Objectives

This work is organized based on three interconnected goals, which are aligned with Zettelkasten principles along the comparative analysis of each RAPL mechanism:

- **Deconstruct implementation challenges** through atomic Literature Notes:

[1] [1] introduced this knowledge-management system; [2] popularized its modern academic applications.
[2] Following [4]'s framework for real-time knowledge organization
[3] See [3], Abstract and Introduction.
[4] See [3], Sections III-IV.

- Overflow correction protocols (MSR vs. powercap) [3, §III-B, §IV-B]
- Timing inaccuracies in polling loops [3, §III-C, §IV-C]
- Domain hierarchy misalignments [3, §III-A]

- **Compare mechanism trade-offs** via interconnected Topic Notes:
  - Privilege requirements (root vs. CAP_BPF) [3, §V-B–V-E]
  - Resiliency across hardware generations [3, §V-F, Table II]
  - Idle-state energy impacts [3, §VI-B2]
- **Formulate implementation guidelines** using Map of Content Notes:
  - Mechanism-to-use-case mapping [3, §VI-D, Table II]
  - Frequency optimization strategies [3, §VI-D, §VII]
  - C-state disruption mitigation [3, §VI-B2, Fig. 10-11]

The analysis implements luhmann1992kommunikation's atomicity principle [1] through granular Literature Notes, while ahrens2017how's connection methodology [2] binds technical insights to practical recommendations in [3, §VII].

2 Background

This section ensures appropriate technical and methodological foundations for tackling a RAPL-based energy measurement analysis, structured through the lens of the Zettelkasten framework of atomic concepts and bidirectional connections.

2.1 Element 1: Software-Based Power Meters Comparison

Recent research offers valuable insights into software-based power meters [5]. Different tools provide varying measurement capabilities:

— **Measurement Granularity**: PowerAPI and Scaphandre offer process-level energy estimation [5]. This extends beyond RAPL's domain-level capabilities. Energy Scope provides system-wide measurements.

— **GPU Support**: Energy Scope and Code Carbon utilize NVIDIA's NVML interface [5]. This enables GPU energy measurement. This complements CPU-focused RAPL mechanisms.

— **Performance Characteristics**: Sampling frequencies vary significantly across tools [5]. Perf and Energy Scope support high frequencies (1000 Hz). Scaphandre is limited to lower frequencies (0.5 Hz). This impacts measurement accuracy.

— **Usability Factors**: Tools differ in ease of use and configurability [5]. Code Carbon and Perf offer user-friendly interfaces. PowerAPI requires more advanced configuration. Documentation quality varies considerably between tools.

These findings complement our RAPL analysis. They provide broader perspectives on energy measurement approaches.

2.2 Element 2: Core RAPL Mechanisms

The Running Average Power Limit (RAPL) provides energy counters for CPU components through four access mechanisms:
— **MSR**: Direct register access requiring root privileges and manual overflow handling [3, §III-B].

— **Powercap**: User-space sysfs interface prone to domain misreporting [3, §III-D].
— **perf-events**: Kernel-managed counters with low-latency polling [3, §V-D].
— **eBPF**: Kernel-space measurement minimizing context switches [3, §V-E].

Key technical challenges include:
— **Overflows**: MSR wraps at 2^{32}, necessitating correction (Table I, [3]).
— **Timing**: Sleep-based polling introduces ±15% jitter at 1000Hz [3, §IV-C].

2.3 Element 3: Zettelkasten Synthesis

Analysis follows three atomic layers:
— **Literature Notes**: 32 atomic insights from [3] (e.g.: "MSR requires CAP_SYS_RAWIO").
— **Topic Notes**: Cross-mechanism patterns (e.g., perf-events' 27W idle advantage over MSR, [3, §VI-B2]).
— **Connections**: Bidirectional links between implementation choices (e.g.: eBPF ↔ C-state disruptions in [3, Fig. 10]).

Methodology aligns with [2]'s principles, enabling traceable synthesis of benchmarks into guidelines.

3 Research Questions

This research aims to dissect software-based methods for measuring CPU energy consumption, focusing on RAPL mechanisms. We formulate the following research questions:

3.1 Main Research Question

RQ.1 How do different RAPL-based mechanisms compare in measuring CPU energy consumption for energy-sensitive computing environments?

This question examines the comparative effectiveness of MSR, powercap, perf-events, and eBPF mechanisms for accessing RAPL. It builds on previous work by [3] and extends recent software-based power meter comparisons from [5]. The focus is on both qualitative differences and quantitative performance impacts.

3.2 Sub-Research Questions

RQ.1.1 What are the qualitative differences between RAPL mechanisms in terms of implementation complexity, required expertise, and system resilience?

This sub-question evaluates the practical aspects of implementing each RAPL mechanism. We will systematically compare the four mechanisms across criteria including technical difficulty, required knowledge, safeguards, privileges, and resilience to architecture changes [3, §V, Table II]. This will provide guidance for selecting appropriate mechanisms based on non-performance requirements.

RQ.1.2 What performance and energy overhead do different RAPL mechanisms introduce during measurement?

This sub-question quantifies the performance impact of each measurement mechanism. We will conduct benchmarks using established testing methodologies like NAS parallel benchmarks [5] across multiple processor architectures. Statistical analysis will determine if certain mechanisms provide significant advantages over others under specific workloads or hardware configurations.

RQ.1.3 How can process-level energy measurement be extended using RAPL-based mechanisms?

Building on the capabilities of tools like PowerAPI and Scaphandre [5], this sub-question explores how RAPL's domain-level measurements can be mapped to process-level consumption.

We will analyze existing approaches for distributing energy measurements among processes and evaluate their accuracy compared to system-wide measurements.

References

[1] Niklas Luhmann. *Kommunikation mit Zettelkästen: Ein Erfahrungsbericht*. Universität Bielefeld, Bielefeld, Germany, 1992. Foundational work on the Zettelkasten method by its creator.

[2] Sönke Ahrens. *How to Take Smart Notes: One Simple Technique to Boost Writing, Learning and Thinking*. CreateSpace Independent Publishing Platform, North Charleston, SC, 2017. Practical guide on using Zettelkasten for academic and creative work.

[3] Guillaume Raffin and Denis Trystram. Dissecting the software-based measurement of CPU energy consumption: A comparative analysis. *arXiv preprint*, arXiv:2401.15985, 2024. Sections referenced: Abstract, Introduction (1); Sections III-IV (2); Section V and Table II (3); Sections III-B and IV-C (4); Sections VI-D and VII (5).

[4] Daniel Schmidt. The zettelkasten method: A framework for real-time knowledge management. *Journal of Information Science*, 46(3):345–360, 2020. Explores Zettelkasten as a tool for dynamic, real-time knowledge organization.

[5] Mathilde Jay, Vladimir Ostapenco, Laurent Lefèvre, Denis Trystram, Anne-Cécile Orgerie, and Benjamin Fichel. An experimental comparison of software-based power meters: focus on cpu and gpu. In *CCGrid 2023 - 23rd IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing*, pages 1–13, Bangalore, India, May 2023. IEEE.

[6] Thomas Müller. Zettelkasten in scientific research: A case study. *Research Methods in the Digital Age*, 12(2):123–140, 2021. Case study on applying Zettelkasten to scientific workflows.

[7] Elena Kaufmann. Digital zettelkasten tools: A comparative analysis. *Journal of Digital Humanities*, 8(1):45–62, 2022. Review of modern tools for implementing Zettelkasten digitally.