Understanding Power Limiting Mechanisms in Modern Processors

A Deep Dive into Intel RAPL and Turbo Boost Dynamics

Romial Menra[1] Guillaume Rosinosky[1] Remous-Aris Koutsiamanis[1]
Sébastien Bolle[2] Jean-Marc Menaud[1]

[1]IMT Atlantique, Inria, LS2N, UMR CNRS 6004, Nantes, France

[2]Orange Research, Meylan, France

Euro-Par 2026

32nd International European Conference on Parallel and Distributed Computing

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Inria

LS2N Laboratoire des Sciences du Numérique de Nantes

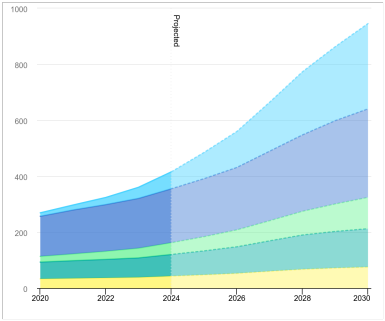
STACK

Outline

1. Motivation
2. How RAPL & Turbo Boost work
3. Related work
4. Problem & contributions
5. An analytical model of turbo duration
6. Experiments & findings
7. Takeaways

Energy is the new bottleneck

- Data center electricity **more than doubles**: about **415 TWh in 2024** to **945 TWh in 2030** [1].
- Broken down by equipment, the **servers** layer, driven by **AI**, grows faster than cooling and network.
- One top machine is already huge: **LineShine** (#1 supercomputer) draws about **42 MW**, the power of a **small town** [2].



Data center electricity (TWh/yr) by equipment, 2020–2030 (IEA Base Case) [3].

The cost is not only energy

- **Carbon**: CO_2 from data center electricity grows from about **180 Mt in 2024** toward about **300 Mt by 2035** [1].
- **Money**: this power is costly, data center demand already pushes up electricity bills, about **$23 bn** added in the US alone (PJM, through 2028) [4].
- **Grid**: in **Europe** the strain is already here, data centers use about **23% of Ireland's electricity**, and **Amsterdam froze new data centers until 2030** for lack of grid capacity [5, 6].

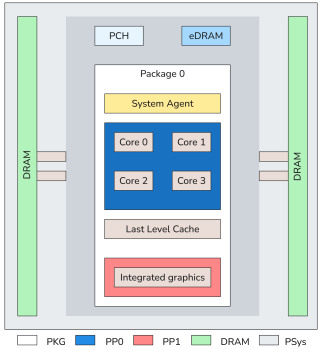
Using **less power** means **less CO_2**, **less money**, and **less strain on the grid**: so we can **limit the power of each machine**.

Power capping: an operational lever

- **Power capping** dynamically restricts the maximum power the hardware may draw, to stay within a system-wide limit [7, 8].
- **Goals:** reduce energy, control heat, and protect the hardware, without losing too much speed.
- **How:** several levers work together, **DVFS** (scale voltage and frequency) and **hardware caps**.
- **This work:** we focus on **Intel** CPUs, where the hardware cap is the **RAPL** interface (*Running Average Power Limit*).

RAPL architecture and power domains

- **History:** proposed by **David *et al.* (2010)** [9], first shipped with **Sandy Bridge (2011)** [10]; on Intel servers ever since.
- RAPL splits the CPU into **power domains** (Package, cores, DRAM, …), accessed via **MSRs** in watts and joules.
- The **package** exposes two **main** limits, each with a **time window** on the order of **ms to seconds**:
  - **PL1** : long-term, window τ_{LT}.
  - **PL2** : short-term, window τ_{ST}.



Four knobs govern the capping behaviour: **PL1**, **PL2**, τ_{LT}, τ_{ST}.

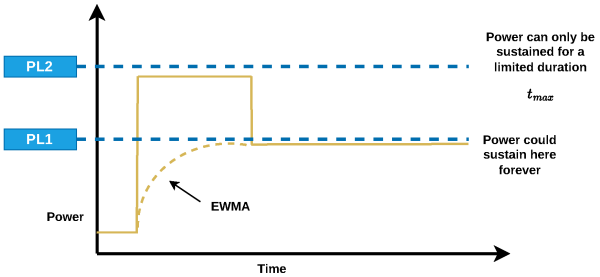
What is Turbo Boost?

- **Definition (Intel):** Turbo Boost lets the CPU cores run **faster than their rated frequency**, as long as they stay within **power and temperature** limits [11].
- **Why:** it speeds up **bursty** or latency-sensitive work when there is power **headroom**.
- **Mechanism:** under load, the package **power rises up to PL2** (the short-term limit); the CPU **cannot hold PL2 forever**.

How RAPL governs Turbo Boost

- Boost duration is set by an **EWMA** * of measured power, tuned by a **Turbo Time Parameter** τ.
- The CPU throttles back once the average reaches **PL1**.
- **We do not know** τ: Intel does not expose it directly.

* Exponentially Weighted Moving Average, documented by Intel [12].



Power held at **PL2** until the average hits **PL1**, after t_{max}; larger $\tau \Rightarrow$ longer boost.

RAPL **enforces** the two limits reliably, but τ, and the **boost duration**, are **not trivially known.**

The problem

- **Setting a limit is easy, choosing it is not:** there are **four knobs**, and nothing tells you what each one does.
- You cannot say **when the server will slow down**, nor whether the cap **saved energy**. Sometimes it costs **more**.

Related work

- **Foundations:** RAPL and the on-chip Power Control Unit were introduced by David and Rotem *et al.* [9, 10].
- **Effectiveness:** hardware capping reacts faster than software [13], but its gains stay workload-dependent [14].
- **HPC trade-offs:** energy–performance has been studied [15, 16], along with relaxing caps [17] and modeling scientific progress [18].
- **Scaling & trust:** RAPL enforces data center **power limits** [19], and its counters were validated against wattmeters [20].

Related work

- **Foundations:** RAPL and the on-chip Power Control Unit were introduced by David and Rotem *et al.* [9, 10].
- **Effectiveness:** hardware capping reacts faster than software [13], but its gains stay workload-dependent [14].
- **HPC trade-offs:** energy–performance has been studied [15, 16], along with relaxing caps [17] and modeling scientific progress [18].
- **Scaling & trust:** RAPL enforces data center **power limits** [19], and its counters were validated against wattmeters [20].

Related work

- **Foundations:** RAPL and the on-chip Power Control Unit were introduced by David and Rotem *et al.* [9, 10].
- **Effectiveness:** hardware capping reacts faster than software [13], but its gains stay workload-dependent [14].
- **HPC trade-offs:** energy–performance has been studied [15, 16], along with relaxing caps [17] and modeling scientific progress [18].
- **Scaling & trust:** RAPL enforces data center **power limits** [19], and its counters were validated against wattmeters [20].

Related work

- **Foundations:** RAPL and the on-chip Power Control Unit were introduced by David and Rotem *et al.* [9, 10].
- **Effectiveness:** hardware capping reacts faster than software [13], but its gains stay workload-dependent [14].
- **HPC trade-offs:** energy–performance has been studied [15, 16], along with relaxing caps [17] and modeling scientific progress [18].
- **Scaling & trust:** RAPL enforces data center **power limits** [19], and its counters were validated against wattmeters [20].

Related work

- **Foundations:** RAPL and the on-chip Power Control Unit were introduced by David and Rotem *et al.* [9, 10].
- **Effectiveness:** hardware capping reacts faster than software [13], but its gains stay workload-dependent [14].
- **HPC trade-offs:** energy–performance has been studied [15, 16], along with relaxing caps [17] and modeling scientific progress [18].
- **Scaling & trust:** RAPL enforces data center **power limits** [19], and its counters were validated against wattmeters [20].

The gap

None of this says **how long a server stays fast** under a cap, and the **energy** studies were run on **older** chips.

Three questions we answer

We break the problem into three concrete questions:

Q1 **What is the impact of each parameter on the turbo duration?**
Four knobs: **PL1**, **PL2**, τ_{LT}, τ_{ST}. Maybe some can be ignored.

Q2 **Can we predict the turbo duration?**
Ideally with a formula, before running anything.

Q3 **Does capping actually save energy?**
Or can it backfire and use more?

Contributions

Three contributions, one per question, all validated on **real Intel servers** across several **microarchitectures**:

A1 **Sensitivity analysis** of the four knobs (**PL1**, **PL2**, τ_{LT}, τ_{ST}): the turbo duration is set by **PL1**, **PL2** and τ_{LT}; the short-term window τ_{ST} has **no measurable effect**, which lets us **deduce the Turbo Time Parameter** τ. (answers Q1)

A2 **Closed-form formula**, from the EWMA logic, that **predicts the turbo boost duration**. (answers Q2)

A3 **Energy-efficiency study**: below **50% of TDP** (Thermal Design Power), capping **raises total energy**. (answers Q3)

From a discrete rule to a closed-form law

1. **Turbo boost recursive EWMA update:**

$$\mathbf{EWMA}_t = \mathbf{EWMA}_{t-\Delta t} + \frac{\Delta t}{\tau}(P_t - \mathbf{EWMA}_{t-\Delta t})$$

2. **Rearranging:**

$$\frac{\mathbf{EWMA}_t - \mathbf{EWMA}_{t-\Delta t}}{\Delta t} = \frac{1}{\tau}(P_t - \mathbf{EWMA}_{t-\Delta t})$$

3. **Limit $\Delta t \to 0$: a first-order ODE ($y = \mathbf{EWMA}$):**

$$\tau \frac{dy}{dt} + y = P(t)$$

4. **Solve** for a constant load $P(t) = \mathbf{PL2}$ from initial power P_{init}**:**

$$y(t) = \mathbf{PL2} + (P_{init} - \mathbf{PL2})\, e^{-t/\tau}$$

5. **Throttle** at $y(t_{\mathsf{max}}) = \mathbf{PL1}$**:**

$$\boxed{t_{\mathsf{max}} = \tau \cdot \ln\left(\frac{\mathbf{PL2} - P_{init}}{\mathbf{PL2} - \mathbf{PL1}}\right)}$$

Experimental setup

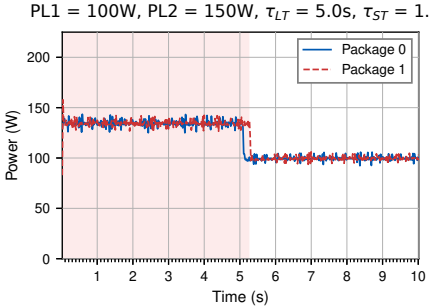
- **Testbed:** SLICES-FR [21]; five Xeon nodes, Skylake-SP → Sapphire Rapids.
- **Measure:** we read the RAPL registers **every 20 ms**; 3-min idle between runs.
- **Loads:** `stress-ng`: `matrixprod` and `fft`.

Node	Microarchitecture (Year)	Sockets	PL1/PL2 (W)	Max PL1 (W)
Dahu	Skylake-SP (2017)	2	70 / 140	125
Roazhon1	Skylake-SP (2018)	2	150 / 250	200
Gros	Cascade Lake-SP (2019)	1	70 / 140	125
Paradoxe	Ice Lake-SP (2023)	2	100 / 150	185
Kinovis	Sapphire Rapids (2024)	2	125 / 190	225

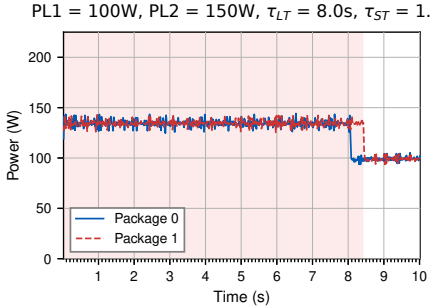
Max **PL1** = the node's **TDP** (Thermal Design Power): the sustained power the cooling is designed to dissipate.

Three phases: sensitivity · $t_{\max}$ validation · energy trade-offs.

Phase 1 — Which knobs matter?



τ_{ST} (500 ms–5 s): **no** effect.

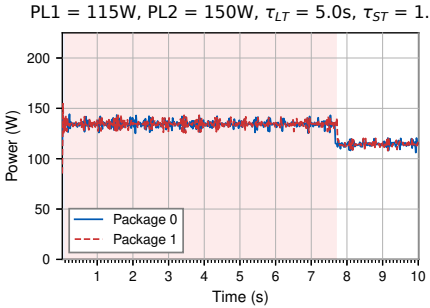


τ_{LT} (5 s→8 s): turbo **extends**.

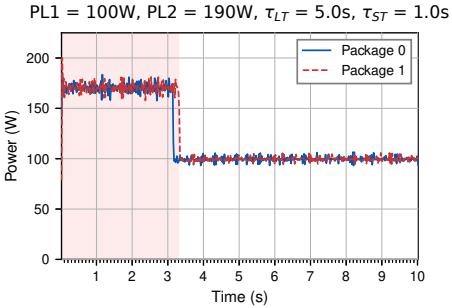
Key idea

τ_{LT} sets how long the turbo lasts; the short-term window τ_{ST} is **negligible.**

Phase 1 — The power limits



PL1↑: longer turbo (**5.3 s**→**7.8 s**).



PL2↑: higher peak, **shorter** turbo.

Synthesis

Turbo duration depends on **PL1**, **PL2**, τ_{LT} only ⇒ map $\tau = \tau_{LT}$.

Phase 2 — Validation on real server

Protocol

- Sweep τ: 250 ms $\rightarrow$ 28 s, $\times 5$.
- Predict t_{max} for two P_{init}:

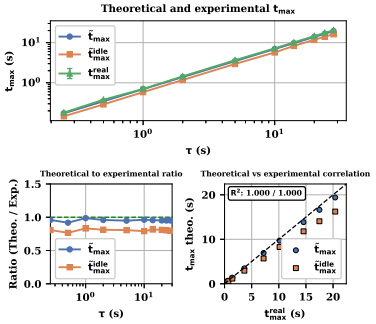
Cold-start ($P_{init} = 0$):

$$\tilde{t}_{\text{max}} = \tau \ln\left(\frac{\text{PL2}}{\text{PL2}-\text{PL1}}\right)$$

Idle ($P_{init} = P_{idle}$):

$$\tilde{t}^{idle}_{\text{max}} = \tau \ln\left(\frac{\text{PL2}-P_{idle}}{\text{PL2}-\text{PL1}}\right)$$

In the plot, $t^{\text{real}}_{\text{max}}$ is the **measured** value.



Gros (Cascade Lake-SP): predicted vs measured t_{max}, $R^2 \approx 1$.

Finding. Predicted t_{max} matches the measurement ($R^2 \approx 1$). The **cold-start** formula ($P_{init} = 0$) predicts best on modern chips (Cascade Lake $\rightarrow$ Sapphire Rapids); only the oldest, **Skylake-SP** (Dahu), prefers the **idle** formula.

Phase 3 — Does capping save energy?

Protocol

- **Turbo off**: both limits pinned to the same cap, **PL1** $=$ **PL2** $= P_{cap}(\alpha) = \alpha \cdot$ TDP.
- α = cap level = the share of the **TDP** left to the package; swept 100% $\rightarrow$ 30%.
- **TDP** (Thermal Design Power) = the sustained power the cooling is designed to dissipate.
- **Fixed work**: 400k ops, the same job at every cap.

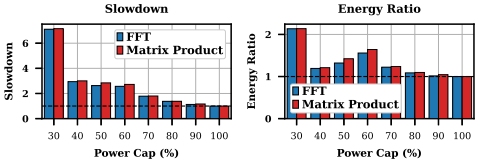
Two metrics, both relative to the uncapped run (100%):

$$\text{Slowdown} = \frac{T(\alpha)}{T(100\%)}, \qquad \text{Energy ratio} = \frac{E(\alpha)}{E(100\%)}$$

For both, **lower is better**; a ratio > 1 means *worse* than uncapped.

Phase 3 — Four observations

1. **Newer is better** for moderate caps (50–100%).
2. **Efficiency collapses** below 50% TDP: energy ratio > 1 (capping costs more total energy).
3. **Workload matters:** `matrixprod` slows down more than `fft`.
4. **Roazhon1**, same Skylake-SP codename as Dahu, delivers the **best energy savings** of the panel.



Dahu (worst case): below 50% TDP, slowdown outweighs the power cut, energy ratio > 1.

Takeaways

Three answers to the RAPL black box

1. **Sensitivity:** turbo is driven by the **long-term window** (τ_{LT}) and the two limits; the **short-term window** (τ_{ST}) is **negligible**.
2. **Model:** a closed-form t_{max} predicts the window with $R^2 \approx 1$ on four microarchitectures: $t_{\text{max}} = \tau_{LT} \ln\left(\frac{\textbf{PL2}}{\textbf{PL2} - \textbf{PL1}}\right)$.
3. **Energy:** capping below 50% of TDP systematically destroys efficiency.

Future work

- Extend to **other workloads**, toward **automatic server profiling**.
- **Distributed** RAPL-based capping across servers under a **global power limit**.

Thank you

Understanding Power Limiting Mechanisms in Modern Processors

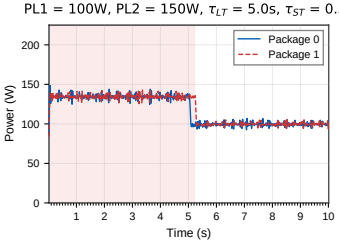
Romial Menra · wedwang-romial.menra@inria.fr

menraromial.com

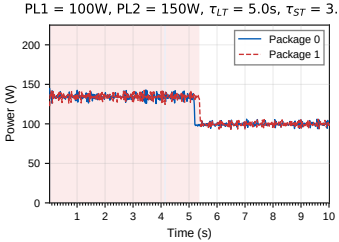
Three things to remember

Know **which knobs matter**, **predict the turbo window** ($t_{\text{max}} = \tau_{LT} \ln \frac{\text{PL2}}{\text{PL2}-\text{PL1}}$), and, **if energy is what matters**, never cap **below 50% of the TDP**.

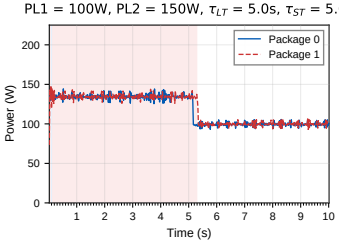
Phase 1: the short-term window τ_{ST} has no effect



$\tau_{ST} = 500\,\text{ms}$



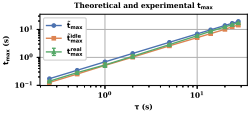
$\tau_{ST} = 3\,\text{s}$



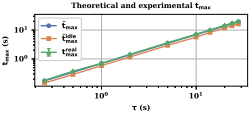
$\tau_{ST} = 5\,\text{s}$

Across the whole τ_{ST} sweep the turbo duration is unchanged (the main slide shows $\tau_{ST} = 1$ s).

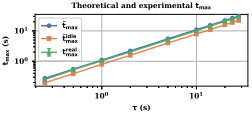
Phase 2: all microarchitectures, $R^2 \approx 1$



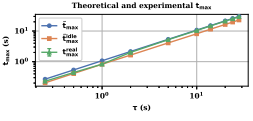
Dahu



Gros

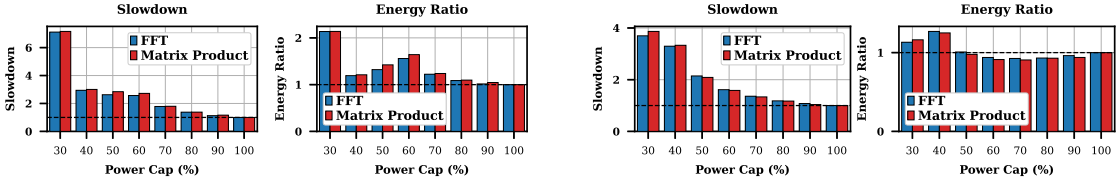


Paradoxe



Kinovis

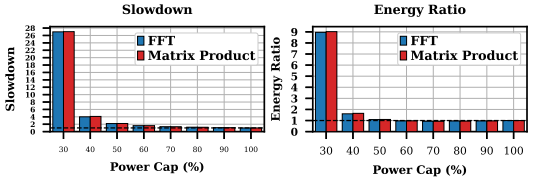
Phase 3: slowdown and energy ratio (1/2)



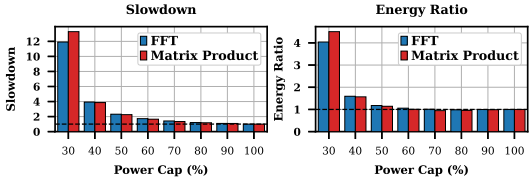
Dahu Roazhon1

Slowdown and Energy ratio vs the cap (**lower is better**).

Phase 3: slowdown and energy ratio (2/2)



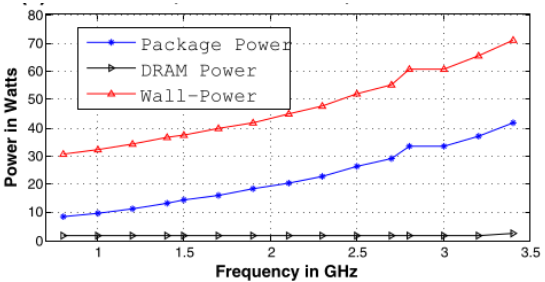
Paradoxe



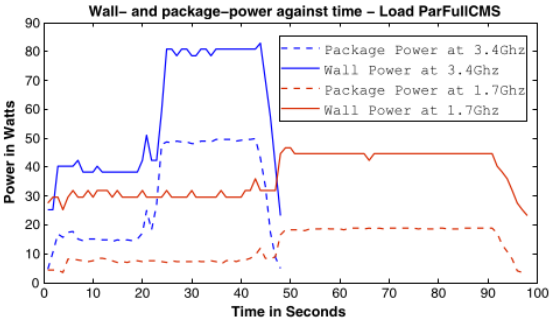
Kinovis

Efficiency collapses below 50% of TDP.

Package power tracks the whole machine's power



Package, DRAM and wall power vs frequency



Package vs wall power over time (two frequencies)

The CPU package power (RAPL) closely tracks the full-machine wall-socket power, so capping the CPU shapes the whole machine's draw [22].

From the recurrence to the exact factor α

Start from the recursive filter, $y[n] = (1-\alpha)\,y[n-1] + \alpha\,P[n]$, with $y \equiv$ EWMA.

1. **Free regime** (load stops, $P = 0$): a geometric sequence,

$$y[n] = (1-\alpha)\,y[n-1] \implies y[n] = y[0]\,(1-\alpha)^n$$

2. **Back to physical time** ($t = n\,\Delta t$, i.e. $n = t/\Delta t$):

$$y(t) = y[0]\,(1-\alpha)^{t/\Delta t} = y[0]\,\left[(1-\alpha)^{1/\Delta t}\right]^t$$

3. **Physical decay** has a time constant τ:

$$y(t) = y[0]\,e^{-t/\tau} = y[0]\,\left[e^{-1/\tau}\right]^t$$

4. **Identify the bases:**

$$(1-\alpha)^{1/\Delta t} = e^{-1/\tau} \implies \boxed{\alpha = 1 - e^{-\Delta t/\tau} \approx \frac{\Delta t}{\tau}}\ (\Delta t \ll \tau)$$

From the recurrence to the differential equation

Same recurrence, now keep the power P:

1. **Error-correction form:**

$$y[n] - y[n-1] = \alpha \left(P[n] - y[n-1] \right)$$

2. **Divide by the step Δt:**

$$\frac{y[n] - y[n-1]}{\Delta t} = \frac{\alpha}{\Delta t} \left(P[n] - y[n-1] \right)$$

3. **Let $\Delta t \to 0$: the left side becomes dy/dt, and $\alpha/\Delta t \to 1/\tau$ (since $\alpha \approx \Delta t/\tau$):**

$$\frac{dy}{dt} = \frac{1}{\tau} \left(P(t) - y \right)$$

4. **First-order ODE:**

$$\boxed{\tau \frac{dy}{dt} + y = P(t)}$$

References I

[1] IEA. *Energy and AI.* Base case: data-centre electricity about 945 TWh by 2030; CO2 from 180 Mt today to 300 Mt by 2035. Paris, 2025. URL: https://www.iea.org/reports/energy-and-ai.

[2] *June 2026 | TOP500.* LineShine, No. 1, 42220 kW. 2026. URL: https://www.top500.org/lists/top500/2026/06/.

[3] IEA. *Global data centre electricity consumption, by equipment, Base Case, 2020-2030.* Licence: CC BY 4.0. Paris, 2025. URL: https://www.iea.org/data-and-statistics/charts/global-data-centre-electricity-consumption-by-equipment-base-case-2020-2030.

[4] Fortune. *Data centers have hiked electricity prices on the public by USD 23 billion.* 2026. URL: https://fortune.com/2026/07/14/data-centers-23-billion-electricity-bills/.

[5] IIEA. *Data Centres in Ireland: The State of Play.* Data centres about 21 per cent of Irish electricity in 2023. 2025. URL: https://www.iiea.com/blog/data-centres-in-ireland-the-state-of-play.

[6] DatacenterDynamics. *The ongoing impact of Amsterdam's data center moratorium.* 2025. URL: https://www.datacenterdynamics.com/en/analysis/the-ongoing-impact-of-amsterdams-data-center-moratorium/.

[7] Arne Tarara. *CPU Power Capping: Processor Energy Configuration Series, Part 3.* Green Coding Solutions. Dec. 2023. URL: https://www.green-coding.io/case-studies/cpu-power-capping/.

References II

[8] Linux Kernel Internals. *Power Capping and RAPL*. URL: https://kernel-internals.org/power/power-capping/.

[9] Howard David et al. “RAPL: Memory power estimation and capping”. In: *International Symposium on Low Power Electronics and Design* (Jan. 2010), pp. 189–194. DOI: 10.1145/1840845.1840883.

[10] Efraim Rotem et al. “Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge”. In: *IEEE Micro* 32.2 (Feb. 2012), pp. 20–27. DOI: 10.1109/mm.2012.12.

[11] Intel. *What Is Intel Turbo Boost Technology and How Does It Work?* Support article 000030893. 2023. URL: https://www.intel.com/content/www/us/en/support/articles/000030893/processors.html.

[12] Intel. *12th Generation Intel® Core™ Processors Datasheet*. Vol. 1. May 2025. URL: https://www.intel.com/content/www/us/en/content-details/655258/12th-generation-intel-core-processors-datasheet-volume-1-of-2.html.

[13] Huazhe Zhang and Henry Hoffmann. “Maximizing performance under a power cap: a comparison of hardware, software, and hybrid techniques”. In: *Proceedings of the 21st International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS ’16)* (Mar. 2016), pp. 545–559. DOI: 10.1145/2872362.2872375.

References III

[14] Pavlos Petoumenos et al. “Power Capping: What Works, What Does Not”. In: *2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS)* (Dec. 2015), pp. 525–534. DOI: 10.1109/icpads.2015.72.

[15] Adam Krzywaniak et al. “Analyzing energy/performance trade-offs with power capping for parallel applications on modern multi and many core processors”. In: *Annals of Computer Science and Information Systems* 15 (Sept. 2018), pp. 339–346. DOI: 10.15439/2018f177.

[16] Azzam Haidar et al. “Investigating power capping toward energy-efficient scientific applications”. In: *Concurrency and Computation Practice and Experience* 31.6 (Mar. 2018). DOI: 10.1002/cpe.4485.

[17] Daniele Cesarini et al. “Benefits in relaxing the power capping constraint”. In: *Proceedings of the 1st Workshop on AutotuniNg and aDaptivity AppRoaches for Energy efficient HPC Systems*. 2017, pp. 1–6. DOI: 10.1145/3152821.3152878.

[18] Srinivasan Ramesh et al. “Understanding the impact of dynamic power capping on application progress”. In: *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE. 2019, pp. 793–804. DOI: 10.1109/IPDPS.2019.00088.

[19] Vladimir Ostapenco et al. *Exploring RAPL as a power capping leverage for Power-Constrained infrastructures*. Jan. 2025, pp. 323–333. DOI: 10.1007/978-981-96-1542-1_19.

References IV

[20] Guillaume Raffin and Denis Trystram. “Dissecting the Software-Based Measurement of CPU Energy Consumption: A Comparative analysis”. In: *IEEE Transactions on Parallel and Distributed Systems* 36.1 (Nov. 2024), pp. 96–107. DOI: 10.1109/tpds.2024.3492336.

[21] Daniel Balouek et al. “Adding Virtualization Capabilities to the Grid’5000 Testbed”. In: *Cloud Computing and Services Science*. Springer, 2013, pp. 3–20. DOI: 10.1007/978-3-319-04519-1_1.

[22] Kashif Nizam Khan et al. “How much power does your server consume? Estimating wall socket power using RAPL measurements”. In: *Computer Science - Research and Development* 31.4 (2016), pp. 207–214. DOI: 10.1007/s00450-016-0325-4.