GPU SMITH
COMPUTE ENGINEERING

GPU SMITH / RESEARCH & PRACTICAL GUIDANCE

LLM Energy Accounting per Tenant: GPU and Server Reconciliation

> Plan and build private AI compute with GPU Smith. We connect workload requirements with GPU hardware, networking, deployment and operating decisions so your infrastructure fits the work it must perform.

Published by GPU Smith · 2026-11-08 · llm energy accounting per tenant · gpu energy attribution · joules per token · nvml energy counter · multi-tenant gpu metering · server power reconciliation · shared gpu cost allocation · continuous batching energy · llm energy per query

Original article: https://gpusmith.com/articles/en/shared-llm-energy-accounting-gpu-joules



In this article

1. Executive Summary
2. Introduction and Background
3. Key Changes in How Shared LLM Energy Is Accounted
4. Implementation Considerations and Process Changes
5. Data Analysis and Evidence
6. Implications and Future Directions
7. Frequently Asked Questions (FAQs)
8. Conclusion

Executive Summary

This report examines how an enterprise private-LLM platform can account for energy per tenant on shared GPUs, and when it should not. The short answer is that a device counter measures a device, not a tenant. NVIDIA's management library exposes a cumulative per-GPU energy counter in millijoules since the driver was last reloaded ((docs.nvidia.com)), supported on Volta or newer devices ((docs.nvidia.com)). Continuous batching then lets the scheduler choose which requests run in each GPU iteration ((www.usenix.org)), so tenant energy is always an allocation of measured totals under a stated policy, never a direct reading.

The policy choice changes the answer materially even when every policy conserves energy. The ML.ENERGY Benchmark derives mean response energy from steady-state energy per token and mean output length ((arxiv.org)), where steady state means saturated batch size. One unreviewed 2026 preprint on request-level attribution, whose main runs used a single GPU and 7B to 14B models, reports that token-proportional allocation differs from an exact Shapley reference by 0.440 normalized L1 on average under static batching ((arxiv.org)) and by 0.458 when the same requests are replayed under continuous batching ((arxiv.org)), while stating that any such reference is counterfactual ((arxiv.org)). In the hypothetical ledger in this report, one 2.70 kWh serving pool gives a single tenant between 0.27 and 0.81 kWh depending on whether requests, output tokens or an explicit weighted rule is used. All three conserve energy; none is measured marginal consumption.

Boundaries matter as much as policies. Google's published methodology reports 0.24 Wh for a median Gemini Apps text prompt on a full-stack boundary ((cloud.google.com)) and 0.10 Wh on a narrower approach ((cloud.google.com)), while the ML.ENERGY paper states that it focuses on software-based GPU energy measurement ((arxiv.org)). A June 2026 preprint that carries GPU-level figures into server-level estimates discloses that CPU, DRAM and networking are not individually modelled ((arxiv.org)). Power usage effectiveness (PUE) is defined against IT equipment energy ((datacenters.lbl.gov)), so multiplying GPU energy by PUE omits the rest of the IT load.

The decision path has three outcomes. Publish an internal tenant report only when GPU counters, a matched server or PDU meter and request logs reconcile within the instruments' own error, with an explicit unknown row. Keep reporting at fleet level when tenant identifiers, cache visibility or window alignment are missing. Collect the missing metering first when the server meter or the request events do not exist. Energy is not emissions: this report makes no carbon-inventory or compliance recommendation.

Introduction and Background

GPU Smith is an independent private AI infrastructure engineering consultancy, and this report is written for platform operators, FinOps (cloud financial management) showback owners and procurement reviewers deciding how far to trust an energy number. The decision is narrow: publish an internal per-tenant energy report, keep reporting at fleet level, or install more metering first. No new measurements were produced; every number in the worked tables is either a cited figure or a labeled hypothetical illustration of bookkeeping arithmetic.

Three claims are often blurred. Operational showback tells a tenant what share of a metered pool it consumed under a published rule. A comparison claim says one model or configuration uses less energy than another under matched conditions. A marginal-energy claim says what the platform's energy would have been without a tenant's requests. The first needs conservation and disclosure, the second needs controlled replay, and the third needs counterfactual evidence that production telemetry does not supply. FinOps guidance states that showback is always required in any FinOps practice ((www.finops.org)), which is a reason to make the showback claim honest, not to avoid it.

The accounting period and tenant boundary must be fixed before any arithmetic. A period is a closed interval with start and end timestamps on a disclosed clock. A tenant is whatever identifier the gateway attaches to each request; whether it is suitable for an aggregate report is a governance question for the platform owner. Energy over an interval is E_J = sum(P_W x interval_seconds) with an explicitly chosen integration rule, or the difference of two reads of a cumulative counter after checking for reset and wrap. Kilowatt-hours are joules divided by 3,600,000. These identities are bookkeeping; they say nothing about causation or fairness.

Key Changes in How Shared LLM Energy Is Accounted

Change 1: The Meter Boundary Comes Before Any Number

Published per-query figures differ by multiples because their boundaries differ. Google states that the energy consumed by idle chips must be factored into its total ((cloud.google.com)); its companion paper puts active accelerator power at 0.14 Wh, or 58% of the 0.24 Wh total ((arxiv.org)). Epoch AI's roughly 0.3 Wh for a typical GPT-4o query ((epoch.ai)) is a modelled figure that folds server and data-center overhead into a per-GPU power assumption ((epoch.ai)), and a 0.42 Wh estimate for a short GPT-4o query is modelled from public API data, not metered power ((arxiv.org)).

A tenant report therefore needs a physical-boundary diagram before it needs a table. The sketch below shows the nested boundaries a ledger must keep apart. It is a conceptual layout, not a measurement.

```
Facility utility meter (total facility energy; PUE numerator)
 +-- Cooling, UPS losses, lighting (outside the IT boundary)
 +-- IT equipment boundary (PUE denominator; metered at UPS, PDU or device input)
      +-- Network switches, storage, other racks (not in the server ledger)
      +-- SERVER AC / PDU OUTLET METER  <-- wall boundary of the tenant ledger
           +-- PSU conversion loss, fans, BMC, NICs, SSDs   (unknown unless metered)
           +-- CPU packages + DRAM  (RAPL counters; check for overlapping domains)
           +-- GPU boards x N  (NVML or DCGM energy counter per device)
                +-- Tenants A, B, C share every iteration  (no per-tenant meter)
```

The tenant ledger lives at the server wall boundary or below. Everything above it is facility overhead, which an operator may apply later to a measured IT boundary but should not blend into device-level numbers.

Change 2: Continuous Batching Breaks the Link Between a Device Counter and a Tenant

Iteration-level scheduling lets the scheduler control how many and which requests run in each model iteration ((www.usenix.org)). In vLLM, completed requests leave and new ones join after each iteration, and batched requests share the same model weights so the overhead of moving weights is amortized ((arxiv.org)). Chunked prefill is enabled by default whenever possible ((docs.vllm.ai)), and the scheduler batches all pending decode requests before scheduling any prefill operations ((docs.vllm.ai)). A single forward pass, and so a single interval of GPU energy, serves several tenants at once.

Whole-device telemetry cannot be split afterward. NVML process-level utilization queries are not supported on MIG-enabled GPUs ((docs.nvidia.com)), and NVML accounting mode is forced to DISABLED there ((docs.nvidia.com)). Those are utilization and accounting interfaces, not power meters. The pages reviewed describe no per-process or per-MIG-instance energy counter, so treat the existence of one as unknown until checked on the exact driver and hardware. Time-slicing adds a limit: DCGM-Exporter does not support associating metrics to containers when GPU time-slicing is enabled ((docs.nvidia.com)). MIG isolates performance resources ((docs.nvidia.com)) with separate paths through the memory system ((docs.nvidia.com)), but that is not an energy meter, and time-slicing provides no fault isolation between replicas ((docs.nvidia.com)).

Open-source attribution projects confirm the problem rather than solve it. The CNCF Kepler documentation states that modern systems lack per-workload energy metering, providing only aggregate readings ((sustainable-computing.io)). For NVIDIA GPUs, Kepler treats power above a per-device baseline as active power ((sustainable-computing.io)), and a June 2026 CNCF post says the re-architected project only attributes active CPU usage per workload ((cncf.io)): even maintained tools make explicit scope choices about idle energy.

Change 3: Allocation Becomes a Named Policy, Not a Measurement

The ML.ENERGY Benchmark is explicit about its target. It submits the whole request dataset up front, excludes ramp-up and ramp-down phases that lack the same energy amortization benefits, and computes average energy per token during the steady state ((arxiv.org)); individual request energy then follows by multiplying that average by each request's output length. This is a defensible estimate for a saturated server. It is not a record of what any one request drew from the wall, and a saturated steady state is not a whole production day: idle, low load, startup and drain fall outside it.

The JouleShare preprint (arXiv identifier 2608.00026, which displays a date of 11 Jul 2026; the displayed metadata is preserved here and its provenance was not independently verified) studies this gap. Titled "Request-Level Energy Attribution for Batched LLM Serving," it targets a charge that reflects each request's average marginal contribution under batching and defines token-proportional attribution as splitting batch energy in proportion to each request's token count ((arxiv.org)). The authors report a mean difference of 0.440 normalized L1 from exact Shapley under static batching ((arxiv.org)) and describe a comparable gap when the same requests are replayed under continuous batching. The main runs used a single data-center GPU, and the authors list validation on live arrival traces as future work ((arxiv.org)). These are one group's reports and were not reproduced here. The narrow supportable point is allocation-target dependence: a rule that conserves batch energy can still disagree with a counterfactual target.

Change 4: Server and Facility Overhead Stay Separate Until a Policy Is Disclosed

Many public per-response figures are GPU-only by design. The ML.ENERGY authors justify GPU-only measurement partly by GPUs' share of provisioned datacenter power ((arxiv.org)), and the v3 analysis notes cases where GPU energy is not just about the GPU because CPU-side bottlenecks can limit batching ((ml.energy)). A 2026 independent preprint cites "ML.ENERGY Leaderboard v3: GPU-level inference energy benchmarks" ((arxiv.org)), uses GPU power as a proxy for full server power, and states that non-GPU server components are not individually modelled. This is observed reuse and a useful boundary question to challenge, not endorsed accounting guidance or a standard multiplier. A USENIX ATC 2025 paper on edge-device LLMs likewise lists the ML.ENERGY leaderboard in its references ((www.usenix.org)), showing that response-level energy figures travel into contexts far from where they were measured.

The accounting consequence is simple. Keep GPU energy, CPU and DRAM energy, and the server-wall remainder as separate rows, choose and disclose an overhead policy later, and apply any facility factor only to a boundary that PUE describes.

Implementation Considerations and Process Changes

Build the Physical Ledger

The ledger starts from meters, not tenants. NVML's energy call returns total GPU energy in millijoules since the driver was last reloaded ((docs.nvidia.com)), and the Zeus library wraps it with a documented unit of mJ ((ml.energy)). DCGM offers a newer field in cumulative whole joules ((docs.nvidia.com)), so a ledger must record which field, unit and truncation was read. Zeus falls back to power polling for GPUs older than Volta ((ml.energy)).

Power readings carry their own semantics. nvidia-smi reports average board power over the last second ((docs.nvidia.com)), supported only on Ampere (except GA100) or newer ((docs.nvidia.com)), while the instantaneous reading is the last measured board-level value ((docs.nvidia.com)). On module-level products the reading covers more than the GPU ((docs.nvidia.com)). Zeus warns that not all GPUs support all power domains and that this is not documented well ((ml.energy)).

For CPU and DRAM, Intel RAPL (Running Average Power Limit) counters are available through Zeus on devices with the interface ((ml.energy)), with root access required ((ml.energy)). Domain overlap is a real hazard. The Linux RAPL driver labels the DRAM counter servers only ((git.kernel.org)), and defines the package counter as consumption of the whole processor package ((git.kernel.org)) and the core counter as consumption of all physical cores ((git.kernel.org)), so core is a subset of package and must not be added to it. Whether package includes DRAM on a given platform is not settled by those lines, and one explainer notes that some RAPL numbers come from estimation or modeling ((hubblo-org.github.io)); verify on the exact CPU and let the wall-meter reconciliation catch the rest. Counters tick in hardware-defined units, such as 1/(2^16) Joules on SandyBridge ((git.kernel.org)), and wrap handling matters because not all hardware sign-extends above the physical width ((git.kernel.org)). As arithmetic only: a 32-bit counter in 1/65,536 J units wraps every 65,536 J, under five minutes at 250 W, so scrape intervals must be shorter than the wrap period of the platform in use.

For the server wall, Redfish defines per-power-supply energy in kilowatt-hour units ((redfish.dmtf.org)), while its older Power schema is deprecated in favor of PowerSubsystem ((redfish.dmtf.org)) and reports average power over an interval, not a counter ((redfish.dmtf.org)). Intel's data center manageability specification treats total input platform power monitoring as an integral expectation ((www.intel.com)) but returns average power over a sampling duration ((www.intel.com)), not an energy counter. Redfish environment metrics also report device-level energy ((redfish.dmtf.org)). Reference benchmarks place analyzers between the wall power and the system under test ((www.spec.org)). A vendor datasheet for one metered rack PDU states IEC plus or minus 1% billing-grade accuracy for energy ((www.eaton.com)), a vendor claim for its own product that serves as an example of an instrument specification to record. A DGX H100 user guide shows the practical shape of the problem: six power supplies in 4+2 redundancy ((docs.nvidia.com)) and a controller supporting Redfish and IPMI ((docs.nvidia.com)), so a server total must be assembled or read at the right point and checked against the PDU.

Table 1 below is the meter-to-tenant ledger. Each row carries an evidence tag: **Measured** (read directly from a meter), **Derived** (computed from totals under a stated policy), **External estimate** (taken from a source the operator did not measure) or **Unknown**.

Ledger row	Source and unit	Window and calibration	Evidence tag
GPU energy, each device	NVML or DCGM energy counter; delta of two reads	Same instants as the server meter; log driver reloads; sensor error for newer GPUs not specified in pages reviewed	**Measured** per device; the sum across all model GPUs is **Derived**
GPU power trace	NVML or nvidia-smi watts	One study found A100 and H100 report a 25 ms average every 100 ms	**Measured**, diagnostics and gap filling only
CPU package and DRAM	RAPL counters per socket	Wrap-checked; package and core domains overlap	**Measured** if non-overlapping, otherwise **Unknown**
Server AC or PDU outlet	Meter kWh or watts; PSU-level Redfish if available	Matched window; clock offset recorded; instrument class from datasheet	**Measured** (reconciliation anchor)
Fans, NICs, storage, PSU loss	Wall minus GPU minus CPU/DRAM	Same window	**Derived** by subtraction; composition **Unknown**
Tenant energy	Pool times policy weight	Per allocation interval	**Derived** under a named policy
Facility overhead	Utility meter, PUE	Different boundary and period; applies to IT energy	**External estimate** unless metered
Idle, startup, drain, maintenance	Counter deltas in logged intervals	Needs runtime state logs	**Measured** energy; segment split **Derived**
Requests with no tenant ID	Gateway logs	Per request	**Unknown** (explicit unallocated row)

Table 1 is more explicit than most showback sheets. The tags keep a measured counter from sitting beside a derived share as if both were readings, and the Unknown row is a first-class entry, not a rounding error. The wall meter is the anchor: everything beneath it must sum back to it, with the remainder shown, not absorbed.

Segment the Accounting Period

Reconcile the full period first, then decide which intervals to allocate. A practical segmentation uses these intervals, each with start and end timestamps from the runtime state log:

- **Startup:** model load and warmup, before the first served request.
- **Saturated serving:** batch size near the configured maximum, the regime benchmarks describe.
- **Low load:** serving with small batches, where static power is spread over few tokens.
- **Drain:** the period after traffic stops while in-flight requests finish.
- **Idle:** model resident, no requests.
- **Failed or cancelled work:** requests that end without a complete response.
- **Maintenance:** restarts, reloads and driver events, which also reset counters.

Segmentation matters because per-token energy is not constant across load. The v3 analysis notes that a GPU consumes static power for the entire execution time ((ml.energy)), and its paper reports that larger batch sizes raise latency, power and throughput while unlocking lower energy per token ((arxiv.org)). A rule calibrated on a saturated interval cannot be applied unchanged to a low-load interval.

Idle policy deserves its own line. Kepler, for instance, splits total energy into active power and a fixed baseline ((sustainable-computing.io)), while other organizations spread idle across tenants. Both are policies. FinOps guidance lists proportional, fixed and even-split models for shared costs ((www.finops.org)), and OpenCost defines cluster idle cost as the portion of allocation costs not allocated to any workload ((www.opencost.io)) and lists shared and idle costs as costs organizations can optionally distribute amongst tenants ((www.opencost.io)). Whichever route is chosen, idle energy must remain visible as its own row.

Choose and Disclose an Allocation Policy

Three simple policies illustrate the choice: equal-request (divide the pool by request count), output-token (divide by generated tokens) and an explicit weighted rule (for example output tokens plus a stated fraction of uncached prompt tokens). For each, allocated_i = distributable_pool x w_i / sum(w), and conservation requires sum(allocated) plus unallocated to equal the measured boundary total for the same interval. OpenCost lists analogous methods for shared costs: proportionate to consumption ((www.opencost.io)) or a custom metric such as bytes of network egress ((www.opencost.io)), and it defines workload cost as the maximum of request and usage, a reserved-capacity rule ((www.opencost.io)). None is fair by default; each moves cost between tenants differently.

The input fields need explicit treatment. A policy document should state at least the following.

- **Prefill versus decode:** Decide whether prompt tokens carry weight; FinOps guidance on proxy metrics allows a variable proportion for shared costs ((www.finops.org)). vLLM exposes engine-wide counters for prompt tokens ((docs.vllm.ai)) and generated tokens ((docs.vllm.ai)); neither is split by tenant, so tenant counts must come from gateway or per-request records.

- **Cached tokens:** Anthropic's usage object separates tokens retrieved from the cache ((docs.anthropic.com)) from tokens written to it ((docs.anthropic.com)), and its input token field excludes both ((docs.anthropic.com)). OpenAI advises tracking the token cache-hit rate by dividing total cached tokens by total input tokens ((platform.openai.com)) and reports cached tokens rounded down to the nearest multiple of 128 ((platform.openai.com)). A cache hit avoids compute that another tenant may have paid for; vLLM notes cache salts to isolate caches in multi-tenant environments ((docs.vllm.ai)) and caches only full blocks ((docs.vllm.ai)). Decide who is credited for shared cache work, and remember that tokens counted by an API field are not joules avoided.
- **Reasoning tokens:** OpenAI states that reasoning tokens are not visible via the API but still occupy context space ((platform.openai.com)), and reasoning models generate far more output tokens per request. A policy that ignores hidden generation understates the tenants that use it.
- **Cancellation and failure:** vLLM records finish reasons including abort ((docs.vllm.ai)). Decide whether aborted work is charged to the requester, shared overhead or Unknown, and apply the rule consistently.
- **Retries and preemption:** A preempted vLLM request is put back in the waiting queue ((docs.vllm.ai)), so one request can consume prefill energy more than once.
- **Requests crossing windows:** The engine timestamps per-request events ((docs.vllm.ai)). Split a long request's tokens across the intervals in which they were generated, or document a convention such as assigning by completion time.

Table 2 below shows what the choice does in a clearly hypothetical single-server hour. It is not a measurement.

Tenant (Hypothetical Example)	Requests	Output tokens	Uncached prompt tokens	Equal-request share (kWh)	Output-token share (kWh)	Weighted share (kWh)
A	20,000	36,000,000	120,000,000	0.54	1.62	1.08
B	60,000	18,000,000	180,000,000	1.62	0.81	0.81
C	20,000	6,000,000	300,000,000	0.54	0.27	0.81
Sum allocated	100,000	60,000,000	600,000,000	**2.70**	**2.70**	**2.70**
Unallocated inside this pool				0.00	0.00	0.00

The distributable pool is 2.70 kWh (9.72 MJ). The weighted policy uses weight = output tokens + 0.1 x uncached prompt tokens, giving weights of 48, 36 and 36 million, or 40%, 30% and 30%. Converted to mean energy per request, tenant A is charged 97.2 J, 291.6 J or 194.4 J, and tenant C 97.2 J, 48.6 J or 145.8 J, depending on the policy. Tenant C's allocation varies threefold (0.27 to 0.81 kWh) with no change in the physical measurement. The columns conserve energy exactly, which is the only property they are guaranteed to have; the meter cannot say which column is right. The 0.1 factor is an arbitrary convention for this example. An operator would have to justify any such factor from its own replay data, and this report does not offer a production-ready weighting.

Enforce Conservation and Expose Residuals

Conservation is checked on the whole boundary, not only the pool. *Table 3* extends the same hypothetical hour into a waterfall from the server meter to tenants.

Waterfall step (Hypothetical Example)	kWh	Evidence tag	Note
Server AC meter, matched window	5.00	**Measured**	Assumed class 1 meter, so plus or minus 0.05 kWh
Sum of 8 GPU energy counters	3.00	**Measured** per device; **Derived** as a sum	Assumed placeholder, not a documented specification: a 5% proportional sensor error would give plus or minus 0.15 kWh
CPU packages and DRAM (non-overlapping domains)	0.60	**Measured**	Overlap checked
Remainder: fans, NICs, storage, PSU loss, unexplained	1.40	**Derived** by subtraction; composition **Unknown**	Carries upstream errors, up to plus or minus 0.20 kWh (meter class plus the assumed GPU term; excludes any CPU/DRAM RAPL error)
GPU energy, saturated serving	2.10	**Measured** (segment deltas)	In the distributable pool
GPU energy, low-load serving	0.60	**Measured**	In the distributable pool
GPU energy, idle	0.20	**Measured**	Unallocated by policy
GPU energy, startup, drain, maintenance	0.10	**Measured**	Unallocated by policy
Allocated to tenants A, B, C	2.70	**Derived** under the weighted policy	Equals the pool in Table 2
Unallocated and unknown total	**2.30**	**Mixed**	0.20 + 0.10 + 0.60 + 1.40
Check: allocated + unallocated	**5.00**		Equals the server meter

The identity 2.70 + 0.20 + 0.10 + 0.60 + 1.40 = 5.00 holds by construction, and that is both the point and the limit. A report showing only the 2.70 kWh would present 54% of wall energy as the whole story; showing the 2.30 kWh of unallocated and unknown energy (46%) lets a reader judge whether the tenant figures are a useful partial view. The remainder row absorbs every error in the three measured rows above it, so its stated uncertainty must combine them. Keep server overhead separate until the operator chooses and discloses a policy, such as spreading CPU, DRAM and remainder by the same weights or holding them as fleet overhead.

Facility overhead is a different step. PUE is the ratio of total facilities energy to IT equipment energy ((datacenters.lbl.gov)), and the Green Grid paper says IT equipment energy should be measured after all facility power conversion, switching, and conditioning is complete ((datacenters.lbl.gov)), at levels from the UPS output ((datacenters.lbl.gov)) to the PDU output ((datacenters.lbl.gov)). Because the IT boundary

includes server power supplies and non-GPU components, multiplying GPU-only energy by a facility factor omits the other IT load. A facility-inclusive figure should apply the factor to the server-wall total (5.00 kWh here) or another measured IT boundary, state which, and stay a separate disclosed row.

Validate Against a Matched Meter and Controlled Replay

Validation has an instrument part and a policy part. For instruments, compare GPU, CPU/DRAM and remainder to the server meter over a window long enough to average out sampling effects. One study found that on A100 and H100 nvidia-smi reports the average of the past 25 ms every 100 ms ((arxiv.org)), so most of the runtime is not sampled, and that following its recommended practice reduced energy error from 39.27% to 4.89% ((arxiv.org)). Those findings concern nvidia-smi power readings on the GPUs tested; whether they carry over to a driver's cumulative energy counter is unknown and must be tested. In general, longer matched windows should tighten agreement and short windows need wider stated bounds.

Window mechanics matter too. Frameworks run GPU computation asynchronously, and Zeus synchronizes CPU and GPU so a window captures all and only the computations dispatched within it ((ml.energy)). Cumulative counters also reset: Triton's energy metric counts since Triton started ((docs.nvidia.com)), and Prometheus documents that counters can only go up and reset when a process restarts ((prometheus.io)). PromQL interprets any decrease between consecutive samples as a reset ((prometheus.io)), so a restart followed by a rise above the previous value within one scrape interval would look continuous; log driver reloads and restarts. DCGM advises recording a baseline before an interval ((docs.nvidia.com)). Request metrics and power metrics also run on different clocks, since Triton states that per-request metrics are unaffected by its interval setting ((docs.nvidia.com)).

Clock alignment sets a floor on precision. RFC 5905 states that clients on fast LANs are within a few hundred microseconds of their servers ((www.rfc-editor.org)), while other conditions leave clients precise within a few tens of milliseconds ((www.rfc-editor.org)). That is small against hour-long windows and large against sub-second request intervals.

Set the acceptance threshold from the actual instruments, not an invented percentage. Take the meter's stated class, note that static-meter accuracy deteriorates under lower load conditions ((www.se.com)), add the GPU-side uncertainty measured on the platform, and require that the unexplained residual be no larger than the combined instrument uncertainty. If it is larger, the report is a finding about metering, not a tenant report. For policy uncertainty, controlled replay is the cleanest tool: replay a fixed request mix with and without a tenant's traffic on the same hardware and compare totals. Any request-level Shapley reference is necessarily counterfactual, and replay results from one model and configuration should not be projected onto others. MLPerf Power's authors acknowledge that some facilities only measure at the PDU level ((arxiv.org)), a useful precedent for documenting granularity limits.

Publish an Evidence Card and a Versioned Policy Change Log

A tenant report should travel with an evidence card containing:

- **Period and clock:** start, end and time source.
- **Meters:** model, unit and calibration for each.
- **Versions:** driver, runtime and power-setting versions.

- **Segmentation:** the rules from the previous section.
- **Policy:** the allocation rule, weights and version.
- **Residuals:** unallocated and unknown rows with uncertainty.
- **Checklist:** the data-quality items below.

Each policy revision gets a version and date, a plain statement of what moved (for example, "prompt tokens now carry weight 0.1"), and a restatement of the previous period under the new rule so tenants can separate policy effects from physical change.

Complete this unknown-data checklist before publication:

- **Hardware:** exact GPU architecture, energy counter support, and the sampling and smoothing interval of any power reading.
- **Coverage:** all GPUs serving the model, including tensor-parallel groups, are in the ledger.
- **Counters:** reset and wrap behavior verified; driver reloads and restarts logged.
- **Clocks:** time source for meters, runtime events and ledger, with measured offset.
- **Runtime events:** per-request start, end, finish reason, token counts, cached tokens and reasoning tokens available per tenant.
- **Shared work:** cache and prefix sharing visible, or declared invisible.
- **Settings:** precision and power-limit settings recorded for each interval.
- **Meters:** accuracy class and low-load behavior documented.
- **Idle and identifiers:** idle policy disclosed; tenant identifiers reviewed for suitability in an aggregate report.

Fleet-level reporting is the only supportable result when any item fails in a way that cannot be bounded: no matched server meter, tenant identifiers absent from request logs, invisible shared-cache work, or an unexplained residual above instrument uncertainty. Whole-device telemetry cannot promise per-process isolation, and the report should not claim it.

Data Analysis and Evidence

The evidence supports a hierarchy. Instruments give totals with documentable error. Benchmarks give steady-state per-token and per-response values on stated hardware. Policies convert totals into tenant shares. Only the first two are measurements.

The ML.ENERGY Benchmark paper (v2, 16 October 2025) reports energy measurements of 40 widely used model architectures across 6 different tasks, on NVIDIA A100 and H100 GPUs, reported at whole-response granularity ((arxiv.org)). The v3 release is a separate, later data set: 46 models across 7 tasks, producing 1,858 configurations on NVIDIA H100 and B200 GPUs ((ml.energy)). In a Qwen 3 32B case study, longer outputs and higher energy per token combine to 23x energy per response for reasoning versus conversation ((ml.energy)), and across models Problem Solving averages 25x more energy per response ((ml.energy)), following the identity that energy per response is energy per token multiplied by output tokens ((arxiv.org)). Because output-length differences dominate, two tenants with identical energy per token can differ by an order of magnitude per response.

Per-query figures from different sources should not be compared directly:

- **Google, median Gemini Apps text prompt (August 2025):** 0.24 Wh comprehensive ((cloud.google.com)), measured from host and accelerator trays based on external power supply units ((arxiv.org)), and Google states the findings do not represent all prompts ((cloud.google.com)). Its narrower figure is 0.10 Wh.
- **Epoch AI (2025):** about 0.3 Wh for a typical GPT-4o query, modelled ((epoch.ai)).
- **Jegham et al.:** 0.42 Wh for a short GPT-4o query, modelled from public API data.
- **ML.ENERGY:** GPU-only steady-state values.

The boundaries are as informative as the values: wall-side tray measurement, modelled cluster power, modelled API-derived estimates and GPU-only steady state are four different quantities. NVIDIA's June 2026 technical blog frames performance per watt as a metric that directly translates to token costs ((developer.nvidia.com)) and states a vendor claim of up to 2.6x more tokens per second per megawatt for an optimized AI factory ((developer.nvidia.com)); that is a facility-scale vendor figure with no per-tenant method.

Instrument error evidence is narrower than many reports imply. NVML documents a plus or minus 5% accuracy statement only for Fermi and Kepler GPUs ((docs.nvidia.com)), so the pages reviewed give no equivalent for newer devices. Independent studies of nvidia-smi sampling and error and a vendor-stated PDU class ((www.eaton.com)) are the documented anchors. Everything else in a reconciliation is the operator's own measurement or a labeled assumption.

Implications and Future Directions

For platform operators, the cheapest improvement is a better ledger, not a better formula: a matched server or PDU meter, per-device counters read as deltas with restart logging, and a runtime that emits per-request records with tenant, token, cache and finish-reason fields. vLLM's own metrics are engine-wide, with prefix-cache hits exposed as counters, so tenant-level token series require gateway instrumentation.

For FinOps and procurement reviewers, the practical question for any energy figure is which rows of Table 1 it covers and what its Unknown row contains. A GPU-only figure can be sound for a GPU-only claim; it becomes a problem when presented as server or facility energy. GPU-level leaderboard values are already being carried into estimates for larger boundaries with disclosed omissions, and a reviewer can ask the same question of any such estimate.

For researchers, the open problems are those the sources flag. Attribution studies so far rely on counterfactual replay of small models on single GPUs, with live-trace validation listed as future work. Multi-GPU, mixed prefill-decode and cache-sharing production traffic is where operators need answers and the evidence is thinnest. Until it exists, claims should stay within what the ledger supports. This report concerns energy only: converting kilowatt-hours to emissions needs separate, disclosed factors and is outside scope, as are compliance conclusions.

Frequently Asked Questions (FAQs)

How should an operator allocate GPU power to multiple tenants? Allocate energy, not instantaneous power: take the counter delta per interval, define a distributable pool, and divide by a disclosed weight such as output tokens, which is one named convention. Keep idle, startup and unknown energy in separate rows.

How can LLM inference energy per request be measured? Instrument a steady serving window, divide energy by tokens produced, and multiply by each request's output tokens, as the ML.ENERGY method does. This is an estimate under a saturated-batch assumption, not an isolated per-request reading.

What is joules per token? Energy over a window divided by tokens generated in it ((arxiv.org)). The value depends on batch size, on whether prompt tokens are counted, and on the boundary (GPU, server or facility).

How does the NVML energy counter work? It returns cumulative GPU energy in millijoules since the driver was last reloaded, on Volta or newer devices (see the ledger above). Read it twice, subtract, and log driver reloads.

How can GPU power be reconciled with server power? Sum all GPU counters, add non-overlapping CPU and DRAM domains, and compare with a matched wall or PDU meter. Carry the difference as a derived remainder row with combined uncertainty (Table 3).

Is shared GPU cost allocation for inference the same as energy allocation? The methods are analogous (FinOps lists proportional, fixed and even-split models and proxy metrics ((www.finops.org))), but cost and energy should not be mixed: a reserved-capacity cost rule can differ from energy consumed.

What is LLM energy consumption per query? Published values range from 0.10 Wh to 0.42 Wh under different boundaries, and Google states its own figure is not representative of all prompts.

Can MIG or time-slicing give multi-tenant GPU power metering? The documents reviewed describe MIG's isolated paths through the memory system ((docs.nvidia.com)) and note that DCGM-Exporter does not associate metrics to containers under time-slicing ((docs.nvidia.com)). They do not describe a per-tenant energy meter.

Conclusion

Per-tenant LLM energy accounting is a reconciliation problem with an allocation step at the end, not a measurement problem. The physical part is measurable: cumulative GPU counters, RAPL domains and a matched server or PDU meter, each with a documented unit, window and error. The tenant part is a policy applied to those totals, and the hypothetical ledger here shows three conserving policies differing threefold for the same tenant over the same hour. A saturated steady-state benchmark gives a defensible per-response estimate but does not describe a full production day, and a token-proportional share is a convention that one group of authors reports differs from a counterfactual reference.

The decision rule is practical. Publish an internal tenant report when the ledger reconciles to a matched meter within combined instrument uncertainty, the unknown and unallocated rows are shown, and the policy and its version are disclosed. Keep reporting at fleet level when tenant identifiers, cache visibility, clock alignment or

the residual cannot be bounded. Collect the missing metering first when the server meter or request events do not exist. In every case, label each number as measured, derived under a stated policy, externally estimated or unknown, and never describe an allocated share as measured marginal consumption.

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. https://docs.nvidia.com/deploy/nvml-api/api/group__nvmlDeviceQueries.html
2. https://www.usenix.org/system/files/osdi22-yu.pdf
3. https://arxiv.org/html/2505.06371v2
4. https://arxiv.org/html/2608.00026v1
5. https://cloud.google.com/blog/products/infrastructure/measuring-the-environmental-impact-of-ai-inference
6. https://arxiv.org/html/2606.10660v1
7. https://datacenters.lbl.gov/sites/default/files/WP49-PUE%20A%20Comprehensive%20Examination%20of%20the%20Metric_v6.pdf
8. https://www.finops.org/framework/capabilities/invoicing-chargeback
9. https://arxiv.org/html/2508.15734v1
10. https://epoch.ai/gradient-updates/how-much-energy-does-chatgpt-use
11. https://arxiv.org/abs/2505.09598
12. https://arxiv.org/pdf/2309.06180
13. https://docs.vllm.ai/en/latest/configuration/optimization
14. https://docs.nvidia.com/deploy/nvml-api/api/group__nvmlAccountingStats.html
15. https://docs.nvidia.com/datacenter/cloud-native/gpu-operator/latest/gpu-sharing.html
16. https://docs.nvidia.com/datacenter/tesla/mig-user-guide/latest/introduction.html
17. https://sustainable-computing.io/kepler/usage/power-attribution
18. https://cncf.io/blog/2026/06/30/kepler-re-architected-improved-power-accuracy-and-a-community-call-to-action
19. https://ml.energy/blog/measurement/energy/diagnosing-inference-energy-consumption-with-the-mlenergy-leaderboard-v30
20. https://www.usenix.org/system/files/atc25-tian.pdf
21. https://ml.energy/zeus/reference/device/gpu/nvidia
22. https://docs.nvidia.com/datacenter/dcgm/latest/reference/field-identifiers.html
23. https://ml.energy/zeus/measure
24. https://docs.nvidia.com/deploy/nvidia-smi
25. https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/plain/arch/x86/events/rapl.c
26. https://hubblo-org.github.io/scaphandre/book/explanations/rapl-domains.html
27. https://redfish.dmtf.org/schemas/v1/PowerSupplyMetrics.v1_1_2.json
28. https://redfish.dmtf.org/schemas/v1/Power.v1_7_1.json
29. https://www.intel.com/content/dam/www/public/us/en/documents/technical-specifications/dcmi-spec.pdf
30. https://redfish.dmtf.org/schemas/v1/EnvironmentMetrics.v1_3_2.json

31. https://www.spec.org/power/docs/SPECpower_ssj2008-Design_Overview.pdf
32. https://www.eaton.com/nz/en-gb/catalog/backup-power-ups-surge-it-power-distribution/eaton-metered-input-rack-pdu.html
33. https://docs.nvidia.com/dgx/dgxh100-user-guide/introduction-to-dgxh100.html
34. https://arxiv.org/html/2601.22076
35. https://www.finops.org/framework/capabilities/allocation
36. https://www.opencost.io/docs/specification
37. https://docs.vllm.ai/en/v0.12.0/design/metrics
38. https://docs.anthropic.com/en/docs/build-with-claude/prompt-caching
39. https://platform.openai.com/docs/guides/prompt-caching
40. https://docs.vllm.ai/en/latest/design/prefix_caching
41. https://platform.openai.com/docs/guides/reasoning
42. https://arxiv.org/html/2312.02741v3
43. https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/user_guide/metrics.html
44. https://prometheus.io/docs/practices/instrumentation
45. https://prometheus.io/docs/prometheus/latest/querying/functions
46. https://docs.nvidia.com/datacenter/dcgm/latest/learn/core-services/fields-and-watches.html
47. https://www.rfc-editor.org/rfc/rfc5905.html
48. https://www.se.com/au/en/faqs/FA276243
49. https://arxiv.org/html/2410.12032v1
50. https://developer.nvidia.com/blog/maximize-ai-factory-energy-efficiency-through-full-stack-inference-and-training-optimizations

THE PUBLISHER

About GPU Smith

GPU Smith provides independent engineering and research for private AI infrastructure. We help technical buyers and operators reason about GPU workload sizing, hardware selection, procurement, deployment and inference operations, from the compute system to the networking, power and cooling requirements around it.

Start with the workload

Useful infrastructure decisions begin with the models, data, throughput, latency and operating constraints a team actually has. GPU Smith helps connect those requirements with system design and deployment choices. The goal is a privately controlled AI environment whose capacity and operational demands are understood before a purchasing decision.

Hardware and supplier research

Our public reference library covers GPUs, complete systems and networking components. The server-vendor directory and manufacturing research help teams investigate suppliers, compare options and follow sources behind technical claims. We distinguish manufacturer specifications from measured benchmarks and advertised capabilities from tested configurations. Reference pages are research resources, not a live inventory listing or a binding equipment quote.

Deployment and operations

GPU Smith's engineering scope includes infrastructure integration, networking, power, cooling and the practical operation of inference workloads. Our published articles explain the tradeoffs behind deployment, procurement and ongoing operations so teams can ask better questions and document decisions.

Work with GPU Smith

Explore the hardware library, GPU references, networking references and vendor research. Contact GPU Smith with your workloads and deployment constraints to discuss a project and confirm current scope, pricing and availability.

Inclusion of a manufacturer or product in our research does not imply a partnership, certification or endorsement.

Website: https://gpusmith.com

This article is educational. Verify consequential medical, legal, financial, regulatory and technical decisions with appropriate professionals and the cited primary sources. Company services are described separately from the article's research findings.