Operational Implications of Dynamic Power Throttling on High-Performance LEO Edge Computing

Gilles Lefranc
EDGX
Ghent, Belgium
gilles.lefranc@edgx.space

***Abstract*—Satellites hosting onboard AI payloads operate under a time-varying power budget set by eclipse, battery state, and concurrent subsystems. A mission planner needs a runtime-adjustable power target whose behaviour is predictable enough to plan against, unlike the coarse, reboot-only factory modes most edge Systems-on-Chip (SoCs) ship with. EDGX's Sterna platform exposes one such interface, Dynamic Power Mode (DPM). This paper asks whether inference throughput on a Jetson Orin NX is a predictable function of the DPM target, and whether that function holds across a full orbit cycle.**

A static power sweep shows that throughput is a linear function of the target, not the cube-root law predicted by classical Dynamic Voltage and Frequency Scaling (DVFS) theory, and that the shape generalises across four Multi-Layer Perceptron (MLP) variants spanning two orders of magnitude in parameter count. An emulated 90-minute orbit cycle with eclipse-phase throttling confirms that queue dynamics follow a fluid-form mass-balance equation, and that a closed-form recovery-window prediction agrees with observation. Matching the target to the workload is more energy-efficient than running unconstrained at equal throughput. The resulting closed-form expressions let a mission planner size queues and schedule target changes from any admissible power schedule without simulation.

***Index Terms*—onboard processing, edge computing, satellite AI, dynamic power management, DVFS, queueing, mission planning, Jetson.**

I. Introduction

LEO SmallSats increasingly carry high-performance AI payloads that process sensor data on board rather than downlinking it raw [1]–[4]. On a shared-power platform, these payloads cannot rely on a constant supply. Eclipse, battery state of charge, and concurrent high-power subsystems (radios, propulsion, thermal) make the usable budget time-varying. Sizing the payload for peak consumption wastes mass and cost; letting it draw any requested power is worse, since the platform may brown out under transient voltage sag, throttle thermally, or be forced to carry battery and solar capacity sized for a worst case that may never occur.

The Jetson Orin family underlies a growing set of in-orbit AI payloads. On the ground, the power and energy consumption of Deep Neural Network (DNN) inference on Jetson has been characterised in detail [5], [6], but always through the vendor-supplied factory power modes (*nvpmodel*). These modes are coarse, discrete, and apply only through a service or device restart, so they cannot serve as a runtime control-plane interface between the platform manager and the payload. Hardware over-provisioning studies in terrestrial High-Performance Computing (HPC) [7] make the case for exposing a fine-grained, software-driven power target instead.

EDGX's Sterna platform exposes such an interface: Dynamic Power Mode (DPM), a runtime-adjustable software power target that the platform manager can reconfigure at any cadence. The controller translates a requested target into an operating point across the voltage, frequency and active-core dimensions. Its internals are out of scope here; we treat DPM as a black-box interface from a requested target to a delivered operating point and ask three operational questions: how predictable the resulting throughput is, what happens to the inference queue when the target drops during eclipse, and what energy is reclaimed by matching the target to the workload.

This paper measures the resulting transfer function and its operational consequences for a spaceborne deployment.

Contributions. This paper:

- derives an empirical transfer function $\mu(P)$ between power target and processing rate for an Multi-Layer Perceptron (MLP) workload on Jetson Orin NX (16GiB), shows it is linear (not cube-root) over the usable 10–25 W operating range, and demonstrates that this shape generalises across four MLP variants spanning 2.6–172 M parameters: the intercept P_0 varies only weakly across variants (5.7–6.4 W) while the slope a scales with work-per-inference. Each variant saturates at its own throughput ceiling, set by a different binding resource (per-inference pipeline overhead, kernel-launch serialisation, or memory bandwidth);
- measures the queue dynamics of a Triton-based inference pipeline with an SSD-backed backlog buffer over a 90-minute orbit cycle and shows that the balance equation $\dot{Q} = \lambda - \mu(t)$ holds to within 1 %;
- derives and validates a closed-form recovery-window formula that a mission planner can evaluate from any $P(t)$ schedule, with a Mean Absolute Percentage Error (MAPE) of 1 %;
- documents an unexpected energy dividend: at equal served throughput, a matched target is $\sim 25\,\%$ more efficient (frames per watt) than an unconstrained operating

point, and explains this quantitatively via a paced-vs-race-to-idle argument.

II. Background

A. Orin NX as an integrated edge AI SoC

The Jetson Orin NX (16 GiB) is a System-on-Module that co-packages an Ampere GPU with eight Cortex-A78AE CPU cores and a shared LPDDR5 memory subsystem on a single die, all delivered through a small number of voltage rails on a common power plane. The same fabricated CMOS process therefore drives both compute domains, so a power-budget knob applied at the module level translates into voltage and frequency moves on multiple clock domains at once. The next subsections introduce the device-physics scaling that governs those moves and the queueing model used later to relate them to throughput.

B. CMOS dynamic power and the DVFS scaling law

Dynamic switching power in a CMOS device is

$$P_{\mathrm{dyn}} = \alpha\, C\, V^2\, f, \tag{1}$$

with α the activity factor, C the switched capacitance, V the supply voltage and f the clock frequency [8]. For compute-bound workloads the processing rate μ is proportional to f. The CMOS delay constraint ties the maximum sustainable clock frequency approximately linearly to the supply voltage ($V \propto f$) [8], collapsing (1) to the textbook cubic law $P \propto f^3$ and a cube-root transfer function $\mu \propto (P - P_0)^{1/3}$.

This nominal scaling rarely holds on modern Systems-on-Chip (SoCs). A Dynamic Voltage and Frequency Scaling (DVFS) controller can only scale voltage down to the process floor $V_{\min}$ at which the transistors remain correct; below that, it can only drop frequency. Over a wide band of the useful operating range, a practical controller pins $V = V_{\min}$ and scales only f, collapsing (1) to $P \propto f$ and therefore $\mu \propto P$ instead of $P^{1/3}$ [9]. Section IV shows that the Orin NX platform operates in this linear regime across the entire range of interest.

C. Queueing the workload

When the processing rate is a function of time (because the power target is a function of time) and the input rate λ is fixed, the instantaneous queue depth $Q(t)$ evolves according to a simple balance equation,

$$\frac{\mathrm{d}Q}{\mathrm{d}t} = \lambda - \mu(P(t)), \tag{2}$$

which is the fluid-flow limit of Little's law [10]. Whether a real system obeys this relation without loss is a separate question: writing and reading backlog data through an SSD-backed queue may introduce discrepancies between the algebraic and the measured in/out rate. This is quantified through two per-phase efficiency coefficients, defined as the ratio of the observed queue rate to the theoretical rate obtained from the simultaneously-measured consumer throughput,

$$\eta_{\mathrm{growth}} = \frac{\dot{Q}^{\mathrm{ecl}}_{\mathrm{obs}}}{\lambda - \mu_{\mathrm{ecl}}}, \qquad \eta_{\mathrm{drain}} = \frac{-\dot{Q}^{\mathrm{rec}}_{\mathrm{obs}}}{\mu_{\mathrm{rec}} - \lambda} \tag{3}$$

where the superscripts ecl and rec refer respectively to the eclipse and recovery phases of the orbit-cycle experiment defined in Section V, and μ_{ecl}, μ_{rec} are the steady-state processing rates measured in each.

III. Methodology

A. Platform

All measurements are taken on a single EDGX Sterna unit built around a Jetson Orin NX (16 GiB). Inference runs on the integrated Ampere GPU via Triton with a TensorRT backend. Two telemetry streams are recorded: the on-device power-rail reading P_{act} at 10 Hz, and CPU and GPU load at 2 Hz. The power target is set through the DPM interface exposed by the platform manager. Only the *measured actual* power P_{act} is reported throughout; the target setpoint is used as an input to the platform and never as a measurement, because the controller's ability to meet it is exactly what is under test.

B. Workload

The primary characterisation workload is a synthetic feed-forward MLP benchmark (`benchmark_mlp`) with $\sim 86\,\mathrm{M}$ parameters, input of shape $(\mathrm{batch}, 1024)$, output of shape $(\mathrm{batch}, 1)$. The model is exported from PyTorch to ONNX in single precision and then compiled to a TensorRT plan with `trtexec --fp16`, so weights and intermediate activations are stored and computed in FP16; client-side request and response tensors remain FP32 over gRPC. The compiled plan is reused across all runs. Batch size is fixed at 64. The workload is deliberately chosen to have no off-device dependencies: dense General Matrix-Matrix Multiply (GEMM) inference on a fixed-shape input, with no filesystem, network or host-side coordination in the hot path.

Three additional MLP variants, *small* (2.6 M params, hidden 512×8), *deep* (43 M, 1024×40) and *wide* (172 M, 4096×10), are used to test workload-invariance of the transfer function and to identify the binding resource at saturation (Section IV-D). Each variant is exported to ONNX with the same opset and compiled to a TensorRT plan with identical trtexec flags (FP16, dynamic batch shapes over 1–512); Phase 1 sweeps are repeated per variant with the same cap schedule, duration and concurrency as for the baseline.

C. Inference pipeline

The payload runs an NVIDIA Triton Inference Server [11] with TensorRT [12] plans as its compute engine. A producer process emits fixed-size batches that feed the server. An on-device SSD-backed ring-buffer queue between producer and server absorbs excess work whenever the processing rate drops below the arrival rate, and preserves batch identity and ordering so that no request is lost.

D. Static characterisation (Phase 1)

The power target is swept from 10 W to 25 W in 500 mW steps, with an additional unconstrained baseline. At each step the pipeline runs for 120 s after a 5 s settling time. A gRPC client generates requests with concurrency of 8 threads to

keep the server continuously backlogged. The client runs in a container on the same Jetson (loopback) so that no external wire limits the observable throughput. Linear fits of $\mu(P)$ are computed over the pre-saturation subset of each sweep. A step is flagged saturated when actual power stops increasing with the target (here above $\sim$22 W for the baseline workload, which removes the 6 highest-target steps of the 32-step sweep from the fit).

E. Dynamic orbit cycle (Phase 2)

A Phase 2 run emulates one 90-minute LEO orbit divided into three 30-minute phases:

- *nominal*, with P_{tgt} chosen so that μ matches a fixed input rate λ;
- *eclipse*, with P_{tgt} reduced to a lower value drawn from the operational envelope (simulating battery-saving);
- *recovery*, with P_{tgt} raised to the maximum allowed by the platform (simulating post-eclipse drain).

The input rate $\lambda = 19\,944\,\text{FPS}$ ($311.6\,\text{batch/s}$ at batch size 64) is matched to the *nominal*-phase μ from Phase 1, so the queue stays near zero in nominal, grows during eclipse, and drains during recovery.

A single in-process runner on the Jetson owns the producer thread, the consumer thread, and the queue. The producer emits fixed-size batches at a deterministic constant-bit-rate cadence ($\lambda/\text{batch} = 311.6\,\text{batches/s}$), so the input stream contributes no arrival-process variance to $\dot{Q}$. The queue is an Non-Volatile Memory Express (NVMe)-backed ring-buffer file (one 256 KiB slot per batch, pre-allocated) with cumulative head and tail offsets, sized above the worst-case eclipse-phase backlog plus a safety margin. Every inferred batch logs its in-system time, queue depth, and the platform's power state.

F. Analysis

Processing rate is reported as steady-state samples/second averaged over the corresponding window, excluding a 30 s settle band after every phase transition. Linear fits use ordinary least squares on the steady-state subset. Phase 1 model comparison tests six candidate functional forms (linear, cube-root, free-exponent power law, quadratic, logistic, Michaelis-Menten), ranked on R^2, residual standard deviation and Bayesian Information Criterion (BIC).

IV. Transfer function $\mu(P)$

A. Observed shape: linear, not cubic

Fig. 1 plots processing rate μ against actual on-device power P_{act} for the benchmark MLP, with three candidate fits overlaid: the theoretical cube-root law, a linear model, and an unconstrained logistic S-curve. The cube-root fit misses the data by thousands of FPS across the entire useful range ($R^2 = 0.63$). The linear and logistic fits are visually indistinguishable below the saturation band and both yield $R^2 > 0.995$; the two-parameter linear model is preferred on BIC.

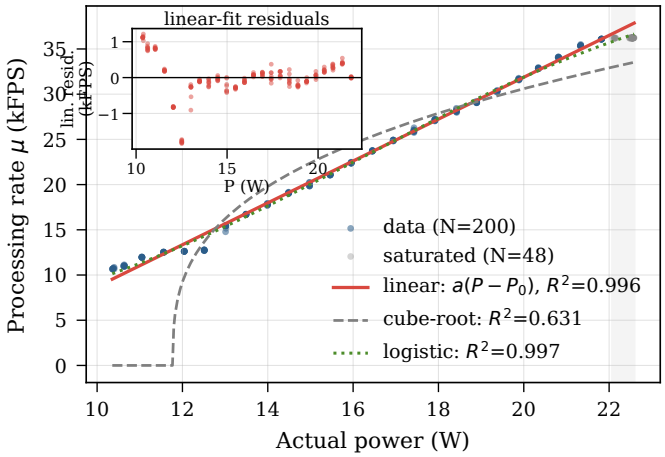

Fig. 1: Processing rate μ vs measured on-device power for the benchmark MLP ($N = 200$ non-saturated points from eight repeat sweeps). Linear fit: $\mu = a(P - P_0)$ with $a = 2.32\,\text{FPS/mW}$, $P_0 = 6.24\,\text{W}$, $R^2 = 0.9955$. Dashed: classical-DVFS cube-root ($R^2 = 0.63$). Dotted: logistic S-curve. Inset: linear-fit residuals.

A formal model comparison across six candidate shapes (linear, cube-root, free-exponent power law, quadratic, logistic, Michaelis–Menten) confirms the ranking. Over the non-saturated range, the cube-root residual standard deviation is 2980 FPS against 544 FPS for the linear model. A free-exponent power-law fit settles at $\beta = 1.22 \pm 0.04$, statistically incompatible with the theoretical $\beta = 1/3$.

The operational form adopted throughout the rest of this paper is

$$\mu(P) = a\,(P - P_0), \qquad a = 2.32\,\text{FPS/mW}, \quad P_0 = 6.24\,\text{W}. \tag{4}$$

P_0 is the extrapolated zero-throughput intercept; it represents the power the device consumes before producing any useful inference (static-power floor of the SoC, rail overhead of active but uncommitted hardware, cost of the service stack). $(P - P_0)$ is then the "useful" power driving the compute pipeline.

Most of the linear fit's residual budget is concentrated in a narrow 11–13 W transition where the platform crosses a discrete operating-regime threshold: an additional CPU core comes online and the GPU clock factor lifts substantially, producing a localised step of $\sim$2 kFPS in throughput. Excluding this transition band tightens R^2 from 0.9955 to 0.9993 without materially shifting the slope or intercept (a shifts by $\sim 2\,\%$, P_0 by $\sim 4\,\%$). Outside the transition, residuals collapse to $\lesssim$500 FPS and match the discrete voltage/frequency steps the DVFS governor traverses as the power target increases. They do not indicate nonlinearity in the underlying relation.

B. Why linear

The cube-root deviation follows directly from how a modern SoC DVFS controller operates. Section II-B noted that $P \propto f^3$ only holds when $V \propto f$. Across the 10–22 W useful range the voltage rail on Orin NX sits at or very near its process floor

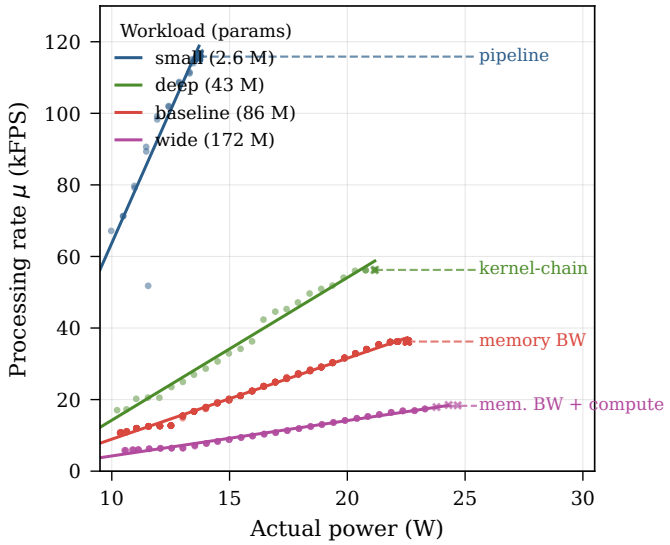

Fig. 2: Per-variant transfer function across the four MLP sizes. Solid lines are linear fits on the pre-saturation data; dashed horizontal segments mark each variant's measured throughput ceiling $\mu_{\text{ceil}}^{\text{wl}}$.

$V_{\min}$, so $P \propto f$ and $\mu \propto P$ follow directly. The $\beta \approx 1.22$ free-fit value suggests a mild super-linearity at the high-power end of the range, consistent with the voltage finally lifting off the floor as the target approaches saturation.

C. Saturation regime

Above $\sim 22\,\text{W}$ the relation flattens: the additional power budget no longer translates into additional throughput, because a non-DVFS ceiling (workload saturation of the GPU tensor pipelines at the chosen batch size) clamps μ. Operationally, a target set above this knee is equivalent to the knee value itself and brings the drawbacks discussed in Section VI-A.

D. Workload generality

The three additional MLP variants (*small* 2.6 M parameters, *deep* 43 M, *wide* 172 M) test whether (4) is a property of the workload or of the platform. Fig. 2 overlays all four variants on a single axis.

The intercept P_0 sits in the narrow band 5.7–6.4 W across all four variants, consistent with its interpretation as a platform-level static-power floor rather than a workload property. The slope a spans $\sim 15\times$ across the variants and scales inversely with parameter count: the small variant delivers the most FPS per watt, the wide variant the fewest, because each additional useful watt delivers a fixed amount of platform work (frequency $\times$ utilisation) and the number of inferences produced per unit work scales inversely with work-per-inference.

The shape $\mu(P) = a(P - P_0)$ therefore generalises across workloads; only the slope a is model-specific and must be measured once per workload, which is all that the mission-planning toolkit of Section V requires. The throughput ceilings in Fig. 2 are a separate observation: each variant plateaus at a different $\mu_{\text{ceil}}^{\text{wl}}$ because it saturates against a different binding resource. These are discussed in Section VI-B.

TABLE I: Orbit-cycle per-window summary, 12 complete runs. The recovery phase is split into an active-drain window (queue > 0) and a post-drain window (queue empty). Predicted μ comes from one of three sources: the producer rate (λ) when the queue is empty, the linear fit (4) when the power target binds, or the workload's throughput ceiling (8) when it binds first.

Window	P_{act} (mW)	μ_{obs} (FPS)	μ_{pred} (FPS)	Duration
nominal	15521 ± 2	19944 ± 1	19944	30 min
eclipse	11604 ± 3	12592 ± 76	12445	30 min
recovery, drain	23303 ± 286	36112 ± 591	36200	810 ± 10 s
recovery, post	19329 ± 155	19944 ± 5	19944	990 ± 10 s

V. Orbit-cycle dynamics

A. Observed cycle

Fig. 3 shows a representative 90-minute orbit cycle in three stacked panels (queue depth, instantaneous processing rate, actual on-device power) with the three programmed phases shaded.[1] Queue depth stays near zero in nominal, grows linearly through eclipse and drains linearly during the active portion of recovery, then sits at zero until the end of the cycle. The processing rate tracks λ in nominal, falls to $\sim 12.6\,\text{kFPS}$ under the eclipse target, rises to $\sim 36\,\text{kFPS}$ during active drain, and returns to λ once the queue is empty.

B. Mass-balance validation across 12 cycles

Across 12 complete cycles (Table I) the per-phase throughputs match the Phase 1 transfer function to within measurement noise. Both the growth slope during eclipse and the drain slope during the active portion of recovery are linear with $R^2 > 0.999$ in all 12 runs.

The per-phase efficiency coefficients defined in (3) are plotted in Fig. 4 (bottom panel):

$$\eta_{\text{growth}} = 1.001 \pm 0.003, \qquad \eta_{\text{drain}} = 0.994 \pm 0.002. \quad (5)$$

Both are within 1 % of unity. The SSD-backed queue layer is therefore throughput-lossless: every sample that enters leaves, and $\dot{Q} = \lambda - \mu(t)$ holds as written, without any correction for queue overhead.

C. Closed-form recovery window

Three properties of the system make a closed-form recovery prediction tractable: (i) $\mu(P)$ is linear below saturation, (ii) $\eta \approx 1$, and (iii) λ is constant. Under these assumptions,

[1] The three-phase profile is an approximation used to test the dynamics of interest (a deficit followed by a recovery window); it does not represent any specific spacecraft's orbit.

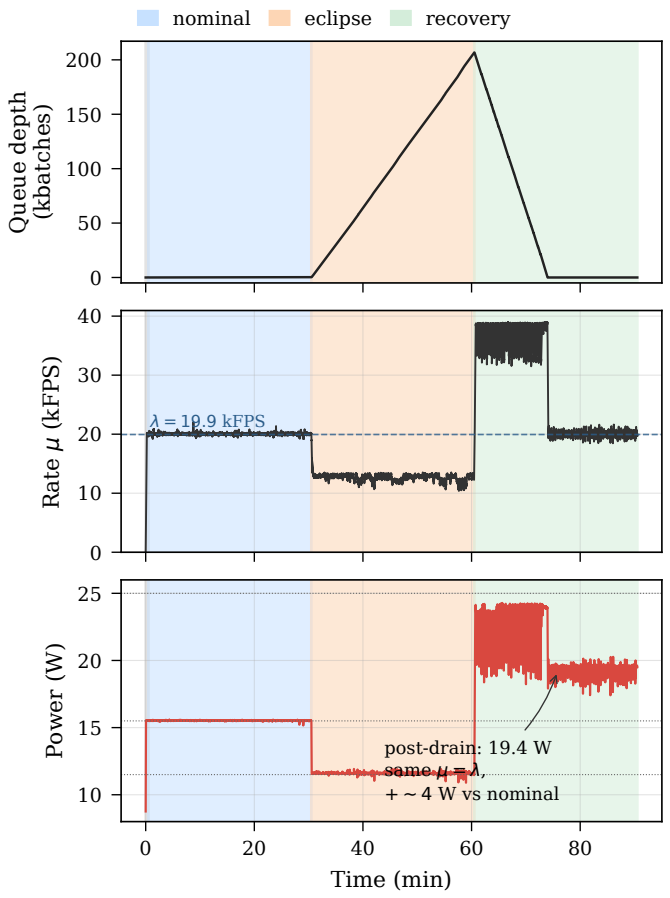

Fig. 3: A representative 90-minute orbit cycle. Top: queue depth $Q(t)$; note how cleanly it grows and drains, with both phases linear to $R^2 > 0.999$. Middle: instantaneous processing rate $\mu(t)$, with the $\lambda = 19.9\,\text{kFPS}$ reference marked. Bottom: actual on-device power, with the three programmed targets shown as dotted lines. The post-drain window, where the device settles at $\sim 19.4\,\text{W}$ rather than the $15.5\,\text{W}$ of the nominal phase despite identical throughput, is a side observation taken up in Section VI-A.

the peak queue depth and drain time admit closed-form expressions:

$$Q_{\text{peak}} = (\lambda - \mu(P_{\text{ecl}})) \cdot t_{\text{ecl}}, \tag{6}$$

$$t_{\text{drain}} = \frac{Q_{\text{peak}}}{\mu_{\text{drain}} - \lambda}, \tag{7}$$

where μ_{drain} is the throughput achieved during the active-drain window. If the linear fit (4) still applies at the recovery target, $\mu_{\text{drain}} = \mu(P_{\max})$. In the more general case the workload hits its own throughput ceiling before the target binds, and μ_{drain} is the roofline-clamped value introduced in Section VI-B. Our baseline workload is in this second regime: $\mu_{\text{drain}} \approx 36.1\,\text{kFPS}$ at an active-drain power of $\sim 23.3\,\text{W}$, matching the Phase 1 saturation plateau at the same operating point.

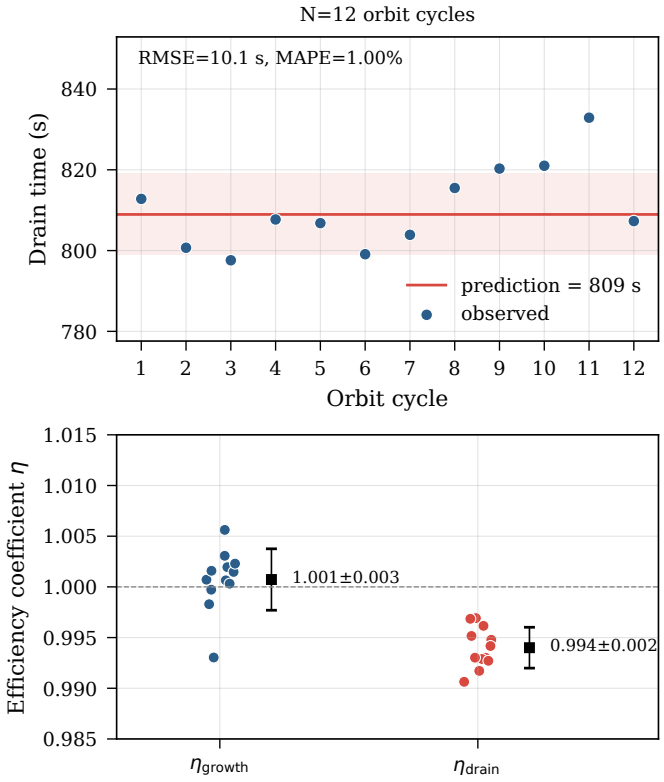

Fig. 4: Top: observed drain time per orbit cycle (12 runs) against the single end-to-end prediction (red line) chaining μ_{ecl}, $\mu_{\text{ceil}}^{\text{wl}}$ and λ through (6) and (7). The shaded band is $\pm$ RMSE. Bottom: per-cycle efficiency coefficients, confirming $\eta \approx 1$ in both growth and drain directions.

Fig. 4 (top) compares the 12 observed drain times to a single end-to-end prediction obtained by chaining the mass-balance inputs through (6) and (7): the eclipse-phase μ_{ecl} sets Q_{peak} over the 30-minute eclipse, and the workload's throughput ceiling $\mu_{\text{ceil}}^{\text{wl}}$ gives μ_{drain} for the recovery phase. The prediction $t_{\text{drain}} \approx 809\,\text{s}$ lands inside the observed cluster $(810.5 \pm 10\,\text{s})$; MAPE is $1.00\,\%$ and Root Mean Square Error (RMSE) is $10\,\text{s}$.

Against a $30\,\text{min}$ recovery budget, the observed t_{drain} (peak queue depth $(206\,590 \pm 2122)$ batches, $13.22\,\text{M}$ samples, $\sim 54\,\text{GB}$ on disk) leaves $16.5\,\text{min}$ of margin. In the passive view this margin tolerates any residual DVFS settling transient and any mission-planning schedule drift. In the active view (taken up in Section VI-A) it is $16.5\,\text{min}$ per orbit of avoidable overhead: the payload is holding an elevated power target that is no longer needed.

VI. Discussion

A. The energy dividend and scheduled target drop

The orbit cycle delivers the same steady-state throughput at two different power levels: $15.52\,\text{W}$ under a matched target and $(19.33 \pm 0.16)\,\text{W}$ with the target left at $25\,\text{W}$.

Fig. 5 quantifies the gap on the throughput-per-watt axis: at equal μ, the matched-target point sits on the platform's upper-bound curve while the loose point sits $25\,\%$ below it.

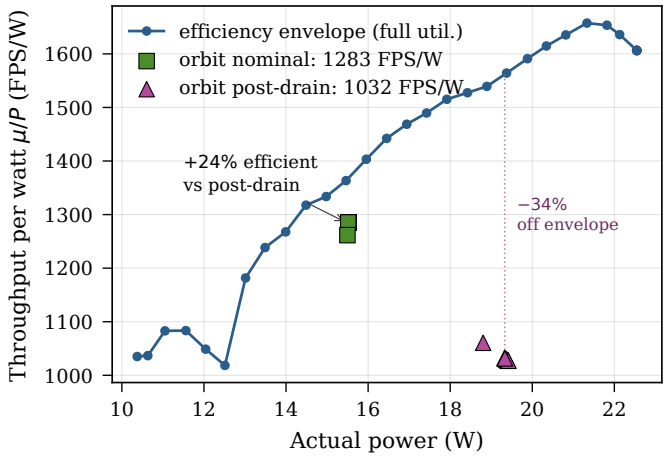

Fig. 5: Throughput-per-watt μ/P vs actual power. The Phase 1 sweep (blue) traces the upper bound: the maximum FPS per watt the platform delivers at full GPU utilisation. Orbit operating points overlaid: nominal (□) sits on this upper bound, post-drain (△) at the same throughput but higher power sits below it. At equal μ, the paced point is 25 % more efficient than the unconstrained one.

The mechanism is the difference in operating regime. Under a matched target DPM holds the device at $\mu(P_{\text{tgt}}) = \lambda$ with (V, f) close to the process floor $V_{\min}$; the GPU is busy 100 % of the time and the operating point sits on the upper-bound curve. Under a loose target the governors give the GPU more power than it needs to keep up with λ. The GPU finishes each batch quickly and then idles between batches. Phase 1 telemetry confirms this mechanism. At the loose target, GPU utilisation drops, and the 3.81 W gap matches the GPU power consumed during that idle fraction. This is the Jetson-scale version of the well-known result that race-to-idle wins only when the idle state is deep [9], [13], [14].

The recovery phase makes this saveable. Because t_{drain} is predictable to 1 % MAPE (Section V), the target can be scheduled to drop from $P_{\max}$ back to $\mu^{-1}(\lambda)$ at $t_{\text{ecl-end}} + t_{\text{drain}}$, reclaiming the 3.81 W gap across the 990 ± 10 s post-drain window. This saves $\sim$3.77 kJ per orbit, or $\sim$0.70 W averaged over the orbit per payload. This figure is specific to the test schedule (one eclipse, λ matched to the nominal target); the saveable margin scales with how often a payload sits above its required target and is bounded above by the 3.81 W gap measured at constant throughput. The schedule is derived from the predicted t_{drain}, so no queue telemetry or detection-to-actuation latency is needed; the 1 % prediction accuracy translates to a worst-case $\sim$25 s error on the scheduled drop, well inside the 990 s window.

B. Roofline coupling: power target vs workload ceiling

Section IV-D established that (4) describes how the *platform* responds to the DPM-controlled power target: slope a scales with work-per-inference and intercept P_0 is workload-invariant. Each variant also saturates at a distinct throughput ceiling $\mu^{\text{wl}}_{\text{ceil}}$. The observed rate follows the standard roofline composition [15]:

$$\mu_{\text{eff}}(P) \;=\; \min\big(\, a(P - P_0),\; \mu^{\text{wl}}_{\text{ceil}} \,\big). \tag{8}$$

Below the knee the power target is what binds, and (4) applies directly. Above it, the workload's own ceiling clamps μ regardless of how much additional power the target permits.

Which resource binds is workload-specific: Fig. 2 showed that the four variants saturate at qualitatively different $\mu^{\text{wl}}_{\text{ceil}}$, with the binding resource ranging from per-inference pipeline overhead (small) through kernel-launch serialisation (deep) to memory bandwidth (baseline, wide). For mission planning it is sufficient to treat $\mu^{\text{wl}}_{\text{ceil}}$ as an empirical per-deployment constant, measured once at $P = P_{\max}$ on the target workload. The first-order scaling of μ_{ceil} with model size and arithmetic intensity is left to a follow-up characterisation.

a) Consequences for Phase 2 predictions: The baseline workload exercises this composition. During active drain the measured power settles at $\sim$ 23.3 W rather than the 25 W target, with throughput clamped at $\mu_{\text{drain}} \approx 36.1$ kFPS by the workload's memory-bandwidth roofline. Plugging the linear extrapolation of (4) at 25 W (43.5 kFPS) into (7) would underpredict drain time by a factor ~ 1.5; using μ_{drain} from (8) recovers the observed ~ 810 s drain and the 1 % MAPE of Section V.

C. A predictive toolkit for mission planners

Combining (4), (6), (7) and (8) the planning workflow reduces to four arithmetic steps given a nominal input rate λ and a power schedule $P(t)$:

1) For each phase i with power target P_i, compute $\mu_i = \min(a(P_i - P_0),\ \mu^{\text{wl}}_{\text{ceil}})$.
2) Queue growth rate in phase i is $\lambda - \mu_i$ (negative means drain).
3) Worst-case queue depth over the schedule is the maximum cumulative integral of (2); for a piecewise-constant $P(t)$ this is a sum of per-phase durations multiplied by per-phase imbalances.
4) Recovery time from any accumulated Q_{peak} is (7); schedule the target to drop from $P_{\max}$ back to $\mu^{-1}(\lambda)$ at $t_{\text{ecl-end}} + t_{\text{drain}}$ to capture the energy dividend of Section VI-A.

The only platform-specific inputs are (a, P_0) from the Phase 1 fit and $\mu^{\text{wl}}_{\text{ceil}}$ measured once per workload. Eclipse buffer sizing and recovery budget trade off through λ and the chosen P_{ecl}.

D. Limitations

The results are measured on a single Sterna unit, mounted on a cold plate held at 30 °C to limit thermal variation across runs, and for a fixed batch size of 64. Three regimes known to affect spaceborne operation are *not* exercised by the presented data.

Thermal throttling: the cold plate keeps the Orin NX well below its throttle threshold, so the clock derating that elevated junction temperature would impose is not visible here. Operation at higher case temperatures would lower the maximum

clock frequency and shift both the slope a and the saturation ceilings reported here.

Workload diversity: the four MLP variants of Section IV-D all run dense FP16 GEMM on fixed-shape inputs, so the empirical claims of this paper are anchored to a single workload class. Verifying that the linear transfer function and the roofline composition extend to convolutional, attention-based or sparse workloads is left for future work; replicating the characterisation on a broader workload mix is the most important follow-up.

Multi-payload contention: the orbit cycle runs one payload. Coordinated target scheduling across multiple Sterna units on a shared bus introduces interactions (arbitration, priority, shared-thermal coupling) that are out of scope for this paper.

VII. Conclusion

Through DPM, the Jetson Orin NX exposes a runtime-adjustable software power target whose effect on inference throughput is more predictable than the classical DVFS model would suggest. Inference throughput is linear in the target, not cube-root: the supply rail sits at its process floor across the useful operating range, so the cubic dependence on frequency collapses to a slope. That fact underwrites the rest of the paper. The mass-balance equation governing an SSD-backed inference queue holds algebraically; the recovery window after an eclipse is computable to within $1\,\%$ from a handful of measured constants; and matching the target to the served throughput is $\sim 25\,\%$ more efficient than running unconstrained, because race-to-idle wins only when the idle state is deep, and Orin NX's is not. Four MLP variants spanning 2.6–$172\,\mathrm{M}$ parameters suggest these claims travel: the linear shape carries across all of them; only the slope and the saturation ceiling are workload-specific.

Together these results yield a predictive toolkit a mission planner can run with a handful of arithmetic operations, taking the linear fit (a, P_0), the workload's saturation ceiling $\mu^{\mathrm{wl}}_{\mathrm{ceil}}$, the producer rate λ and an orbit power schedule as inputs. The operational consequence is that the lowest target sustaining λ also minimises power at that throughput, and the predicted t_{drain} fixes the moment at which the recovery target can be dropped. This lets a payload reclaim power that an unconstrained operating point would otherwise spend on shallow idle, and the saved margin aggregates directly into the Electrical Power System (EPS) budget on a multi-Sterna bus.

Future work should test the characterisation on convolutional, attention-based and sparse workloads, extend it to multi-payload coordination on a shared bus, and verify the behaviour under elevated case temperatures where thermal throttling lowers the maximum clock frequency.

References

[1] G. Furano *et al.*, "Towards the use of artificial intelligence on the edge in space systems: challenges and opportunities," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 35, no. 12, 2020.

[2] G. Giuffrida *et al.*, "The Φ-Sat-1 mission: the first on-board deep neural network demonstrator for satellite Earth observation," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022.

[3] R. Pitonak *et al.*, "CloudSatNet: A lightweight deep learning model for on-board cloud segmentation on CubeSats," *Remote Sensing*, vol. 14, no. 13, 2022.

[4] M. Ghiglione and V. Serra, "Opportunities and challenges of AI on satellite processing units," in *Proc. ACM CF*, 2022.

[5] A. Ma *et al.*, "Performance, power, and energy analysis of deep neural networks on NVIDIA Jetson platforms," in *Proc. IEEE HPEC*, 2021.

[6] L. Suarez *et al.*, "Power characterization of edge AI accelerators for space applications," *Acta Astronautica*, 2023.

[7] T. Patki *et al.*, "Exploring hardware overprovisioning in power-constrained, high performance computing," in *Proc. ACM ICS*, 2013.

[8] N. H. E. Weste and D. M. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 4th ed. Addison-Wesley, 2010.

[9] E. Le Sueur and G. Heiser, "Dynamic voltage and frequency scaling: the laws of diminishing returns," in *Proc. HotPower*, 2010.

[10] J. D. C. Little, "A proof for the queuing formula $L = \lambda W$," *Operations Research*, vol. 9, no. 3, pp. 383–387, 1961.

[11] NVIDIA Corporation, "Triton Inference Server documentation," 2024.

[12] NVIDIA Corporation, "TensorRT developer guide," 2024.

[13] F. Yao, A. Demers and S. Shenker, "A scheduling model for reduced CPU energy," in *Proc. IEEE FOCS*, 1995.

[14] S. Albers and A. Antoniadis, "Race to idle: new algorithms for speed scaling with a sleep state," *ACM Trans. Algorithms*, 2014.

[15] S. Williams, A. Waterman and D. Patterson, "Roofline: an insightful visual performance model for multicore architectures," *Communications of the ACM*, vol. 52, no. 4, pp. 65–76, 2009.