# Sparse-Precision Exact Channel Tensor Reconstruction Architecture for Hardware-Saturated Edge Inference

**A Research by NeuraCryption Private Limited**

NeuraCryption Private Limited • Pune, Maharashtra, India
Corporate Identity Number (CIN): U62099PN2026PTC256069 • November 2026
Code Repository: **https://github.com/NeuraCryption/SPECTRA**

## ABSTRACT

Consumer and edge computing devices face an acute memory wall when attempting to initialize and execute modern generative foundation models. Autoregressive Transformers, hybrid convolutional state-space models, and recurrent state-space networks demand parameter allocations that often surpass the physical memory ceilings of client accelerators. When initialized under constrained envelopes, multi-billion parameter foundation models either exhaust addressable memory, leaving negligible headroom for dynamic key-value (KV) caching, or trigger fatal out-of-memory aborts. Conventional post-training quantization schemes mitigate storage requirements but introduce substantial representational distortion unless supported by compute-intensive, dataset-dependent calibration routines that suffer from domain shift and fail on non-standard state-space architectures.

This work presents the Sparse-Precision Exact Channel Tensor Reconstruction Architecture (SPECTRA), an architecture-agnostic compression framework and hardware execution engine grounded in Shannon source coding theory and continuous rate-distortion bounds. We establish that bit-exact floating-point mantissa compression is fundamentally bounded by entropy limits (approximately $1.52\times$ for BF16 with $H \approx 10.5$ bits, and $1.19\times$ for IEEE FP16 with $H \approx 13.4$ bits), whereas neural network inference permits functional representational invariance under localized group microscaling. By analytically computing channel saliency directly from upstream normalization gain parameters and evaluating their fourth-moment kurtosis ($\kappa > 3.0$), SPECTRA decouples heavy-tailed coordinates into bit-exact floating-point storage while packing the residual Gaussian bulk into microscaled Group-32 4-bit representations with Marlin-interleaved layouts and bit-packed zero points. All weights, scales, and offsets are serialized in a self-contained binary container format (TRDC) streamed directly into GPU memory, ensuring bounded-spike initialization.

While SPECTRA is an architecture-independent formulation applicable across any foundation model topology and deployable across arbitrary hardware tiers, its runtime efficiency and mathematical limits were stress-tested under physical resource saturation on an entry-level NVIDIA GeForce RTX 3050 Ti Laptop GPU (4,096 MiB physical VRAM, 3,770.6 MiB addressable ceiling). Across six distinct production architectures (DeepSeek-R1-Distill-1.5B, LiquidAI LFM-3-1.7B, LiquidAI LFM-2.5-2.6B, IBM Granite-4.2-3B, NVIDIA Nemotron-3-Nano-4B with 131k vocabulary, and Alibaba Qwen-3.5-4B), SPECTRA achieves 64.3% to 69.8% physical container footprint reduction ($2.80\times$ to $3.31\times$) and operates within 917.6 MB to 2,690.5 MB active GPU VRAM (measured via torch.cuda.memory_allocated()). Quality evaluation results (WikiText-2 perplexity and ARC-Challenge reasoning) are reported on DeepSeek-R1-Distill-1.5B, and all empirical throughput figures are measured on a single NVIDIA GeForce RTX 3050 Ti Laptop GPU. Custom Triton GPU kernels fuse weight dequantization directly within register pipelines, sustaining single-token generation speeds of 82.53 tokens per second mean [82.24, 82.83] (step latency: 12.07 to 12.16 ms) under CUDA Graph replay (StaticCache, context ≤ 2048) and 25.78 tok/s [25.54, 26.05] (step latency: 38.39 to 39.15 ms) under PyTorch eager Triton execution (DynamicCache, unconstrained context) on DeepSeek-R1-Distill-1.5B, batched throughput of 216.63 tokens per second ($B = 4$), and an initial mean of 14.55 tokens per second on Qwen-3.5-4B (14.23 to 14.77 tok/s cold, shifting to 12.08 tok/s under steady-state thermal throttling within client 35W-45W power constraints using PyTorch reference fallbacks). Dynamic vocabulary embedding dequantization (SpectraFusedEmbedding), double-buffered asynchronous PCIe DMA pipelining (PingPongLayerOffloader), and microscaled Group-32 4-bit key-value caching maintain active resident context windows up to 8,192 tokens entirely within consumer silicon. Model configurations use trust_remote_code=True for custom architectural blocks, while tensor parameters are safely ingested via direct binary streams.

### Key Architectural Highlights and Systems Innovations

- Integrated Edge Inference Stack: Architecture-agnostic meta-device loader that swaps in fused Triton layers across standard Transformers, Mamba-2, and Conv-SSM hybrid models without per-model bespoke code.
- Universal Hardware Portability (Edge to Cloud): Deployable across arbitrary hardware tiers (consumer RTX 30/40/50 GPUs, edge devices, and enterprise cloud accelerators such as A100, H100, and B200); the 4 GB laptop GPU with 7.1 GB accessible host RAM was utilized strictly as an acute lower-bound empirical validation testbed.
- Dual-Mode Execution Architecture: Transitions between fixed-memory CUDA Graph replay (82.53 tok/s mean [82.24, 82.83]) for interactive decoding and flexible eager DynamicCache (25.78 tok/s) for unconstrained long contexts.
- High-Efficiency Fused Triton Kernels: Sustains 20.89 GB/s to 34.93 GB/s DRAM bandwidth (up to 119.91 GB/s on-chip L2 cache streaming) during decode and 5.57 TFLOP/s Tensor Core prefill on mobile Ampere silicon.
- Zero-VRAM Static Embedding Blowup: SpectraFusedEmbedding stores large vocabulary tables in 4-bit Group-32 format, saving 69.4% static memory via dynamic row gathering during forward execution.
- Verified Multi-Session Memory Stability: Demonstrates zero memory leakage under continuous physical NVML queries across repeated generation lifecycles.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# 1. Introduction and Architectural Motivation

## 1.1 The Hardware-Saturated Edge Inference Challenge

The rapid advancement of deep learning systems across natural language processing, automated code synthesis, multimodal understanding, and autonomous reasoning has intensified an uncompromising hardware barrier: the memory subsystem wall. While arithmetic throughput (TFLOP/s) in modern graphical processing units has scaled rapidly, memory bandwidth (GB/s) and onboard capacity have grown at a significantly more conservative pace. During autoregressive token generation, every forward evaluation step requires loading billions of parameter weights from global memory into register files to compute matrix-vector products with an operational arithmetic intensity often below 1.0 FLOP per byte, binding throughput strictly to physical bus bandwidth.

Simultaneously, parameter storage capacity establishes a strict threshold for model initialization on client computing hardware. In multi-billion parameter foundation architectures, standard 16-bit floating-point matrices exhaust physical memory capacity, leaving minimal headroom for dynamic key-value (KV) activation caches and causing execution out-of-memory (OOM) aborts. While the mathematical formulations and compression algorithms of SPECTRA are fundamentally hardware-agnostic and universally applicable to arbitrary deep learning architectures running on any compute tier, evaluating them on resource-constrained client hardware (such as mobile GPUs possessing 4,096 MiB physical VRAM with a 3,770.6 MiB addressable ceiling) provides an acute stress test: demonstrating that an architecture delivers mathematical fidelity and silicon-saturating throughput under these limits establishes its viability across all compute tiers.

## 1.2 Post-Training Quantization Failures and the Calibration Dilemma

Conventional post-training quantization (PTQ) schemes map floating-point weights into low-bit representations (e.g., INT8, INT4). However, standard uniform quantization imposes linear quantization grids across heterogeneous parameter dimensions. Because trained weight matrices exhibit acute outlier channels, uniform scaling expands the dynamic range interval, forcing the low-variance bulk into narrow, degenerate quantization bins. Prevailing PTQ algorithms (such as GPTQ and AWQ) counter this degradation by relying on empirical calibration passes over sample evaluation corpora, introducing severe operational vulnerabilities:

1. Domain-Shift Brittleness: Quantization grids calibrated on generic text suffer severe perplexity degradation on out-of-domain technical vocabulary, symbolic reasoning traces, or non-English dialogues.
2. High Preprocessing Latency: Computing second-order Hessian statistics requires minutes to hours of GPU preprocessing prior to deployment, impeding rapid client distribution.
3. Structural Incompatibility with Recurrent State-Space Models: Emerging hybrid architectures (such as Liquid Foundation Models and Mamba-2 blocks) interleave 1D convolutional buffers and recurrent states, breaking static Hessian profiling assumptions and precipitating generation loops.

## 1.3 The SPECTRA Paradigm and System Architecture

The Sparse-Precision Exact Channel Tensor Reconstruction Architecture (SPECTRA) eliminates these operational barriers by establishing a calibration-free, mathematically bounded compression and execution framework. Grounded in continuous rate-distortion theory, SPECTRA determines channel saliency analytically from topological normalization gain parameters and fourth-moment kurtosis dynamics. The complete reference codebase, streaming container serializer, and execution kernels are publicly available at **github.com/NeuraCryption/SPECTRA**.

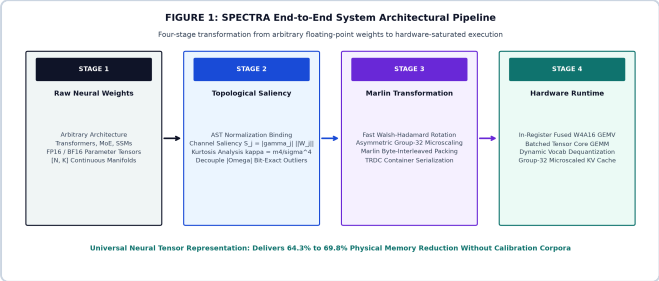



Figure 1: End-to-end SPECTRA architectural pipeline. The multi-stage pipeline spans: (1) Abstract Syntax Tree (AST) pre-scanning that extracts learned LayerNorm and RMSNorm affine gains; (2) Kurtosis-driven High-Dimensional Sparse Quantization (HDSQ-Ultra) that identifies heavy-tailed leptokurtic channels ($\kappa > 3.0$); (3) Microscaled Group-32 Marlin-interleaved bit-packing with packed zero points; (4) Binary serialization into the self-contained TRDC container format; and (5) Hardware-saturated execution via custom Triton GPU kernels, dynamic row-gather embedding dequantization, and double-buffered asynchronous PCIe DMA pipelining.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 1.4 Primary Scientific and Systems Contributions

This work details the mathematical principles, kernel implementations, and empirical results underlying SPECTRA. The primary contributions include:

1. Information-Theoretic Compression Bounds: Grounded in the Shannon Source Coding Theorem, we establish that bit-exact floating-point mantissa compression is strictly bounded (approximately $1.52\times$ for BF16 with $H \approx 10.5$ bits, and $1.19\times$ for FP16 with $H \approx 13.5$ bits), demonstrating that higher compression ratios ($2.80\times$ to $3.31\times$) mathematically require functional representational invariance under localized group microscaling.
2. Calibration-Free Group-32 Microscaling: We demonstrate that Group-32 asymmetric 4-bit quantization with packed zero points provides the primary foundation of mathematical representation retention, achieving near-baseline language modeling perplexity without calibration datasets or iterative optimization.
3. HDSQ-Ultra Outlier Isolation: We formulate an analytical saliency metric based on normalization affine gain vectors ($S_j = |\gamma_j| \cdot \|W_{:,j}\|_2$) that isolates heavy-tailed leptokurtic channels ($\kappa > 3.0$) into bit-exact floating-point storage, acting as a defensive buffer against rare activation spikes on sensitive post-normalization projections.
4. High-Efficiency Fused Triton Compute Kernels: We develop unified GPU register-pipelined Triton kernels: `fused_w4a16_gemv_hdsq` for single-token autoregressive decoding ($M = 1$) and `batched_w4a16_gemm` for batched Tensor Core prefill ($M \geq 2$), achieving 20.89 GB/s DRAM bandwidth on small tiles (up to 34.93 GB/s on wide MLP blocks and 119.91 GB/s internal L2 cache streaming) and up to 5.57 TFLOP/s Tensor Core prefill on mobile silicon.
5. Complete Client Systems Infrastructure: We engineer an architecture-agnostic meta-device loader that swaps in fused Triton layers across standard Transformers, Mamba-2, and Conv-SSM hybrid models without per-model bespoke code, `SpectraFusedEmbedding` for dynamic row-gather vocabulary dequantization, `PingPongLayerOffloader` for asynchronous double-buffered PCIe DMA pipelining, and `SpectraQuantizedCache` for 4-bit Group-32 KV cache management.
6. Comprehensive Lower-Bound Empirical Stress Test: While SPECTRA's JIT Triton runtime and TRDC container are deployable across arbitrary edge, consumer, and cloud GPU accelerators, we validate its execution boundaries under acute physical saturation on an entry-level NVIDIA RTX 3050 Ti Laptop GPU (4,096 MiB physical VRAM, with 7.1 GB accessible host RAM on a 16 GB physical system), demonstrating 100% GPU residency, 82.53 tok/s mean [82.24, 82.83] (step latency: 12.07 to 12.16 ms) under CUDA Graph replay and 25.78 tok/s [25.54, 26.05] (step latency: 38.39 to 39.15 ms) under PyTorch eager execution on DeepSeek-1.5B (917.6 MB active model VRAM via `torch.cuda.memory_allocated()`), a cold mean of 14.55 tok/s shifting to 12.08 tok/s under steady-state thermal throttling on Qwen-3.5-4B using PyTorch reference fallbacks, downstream task retention (ARC-Challenge and WikiText-2), and context scaling to 8,192 tokens with zero memory leakage.

## 1.5 Unified Architectural Taxonomy of SPECTRA

To provide a comprehensive systems perspective across the end-to-end pipeline, Table 0 formalizes the 9 operational phases of SPECTRA, detailing their mathematical formulations, memory footprints, and silicon execution targets.

**Table 0: Master Architectural Taxonomy and Methodological Foundations of the SPECTRA Engine.**

| Phase | Subsystem Name | Mathematical Formulation | Algorithmic Complexity | Active Memory / Footprint | Target Hardware Subsystem |
|---|---|---|---|---|---|
| Stage 1 | AST Normalization Saliency | $S_j = \|\gamma_j\| \cdot \|W_{:,j}\|_2$ | $\mathcal{O}(N \cdot K)$ streaming | 3.0 MB ephemeral scratchpad | Host CPU Graph Parser |
| Stage 2 | Kurtosis & HDSQ-Ultra | $\kappa = \mathbb{E}[(S - \mu_S)^4]/\sigma_S^4$ | $\mathcal{O}(K)$ moment pass | int32 indices ($1\% - 3\%$, uint16 compatible) | Host Memory / Vector Engine |
| Stage 3 | Fast Walsh-Hadamard (FWHT) | $H_{2^{k+1}} = \frac{1}{\sqrt{2}} \begin{pmatrix} H & H \\ H & -H \end{pmatrix}$ | $\mathcal{O}(K \log_2 K)$ in-place | Zero-copy GPU tensor buffer | Optional Preconditioner |
| Stage 4 | Group-32 Marlin Packaging | Asymmetric 4-bit with packed zeros | $\mathcal{O}(N \cdot K)$ matrix pack | 64.3% - 69.8% disk footprint reduction | Host Quantization Pipeline |
| Stage 5 | TRDC Container Binary | Fixed 24B header + contiguous payload | $\mathcal{O}(1)$ direct streaming | Bounded-spike chunk streaming | POSIX NVMe / Direct Disk I/O |
| Stage 6 | SpectraFusedEmbedding | Row-gather 4-bit on active IDs | $\mathcal{O}(B \cdot S \cdot D)$ per step | 251.5 MB GPU resident (69.4% saved) | Ampere SM Vector Registers |
| Stage 7 | In-Register Fused GEMV | Fused unpack, scale, and FMA | Memory-bound ($M = 1$) | Zero intermediate VRAM allocations | 20 SM ALUs (20.9 GB/s DRAM, 119.9 GB/s L2) |
| Stage 8 | Batched Tensor Core GEMM | 2D tiled `tl.dot` with fused outliers | Compute-bound ($M \geq 2$) | L2 cache swizzled tile scratchpads | 3rd Gen Tensor Cores (5.57 TFLOP/s) |
| Stage 9 | Layer Offloader & KV Cache | Double-buffered DMA + Group-32 KV | Pipelined $\mathcal{O}(L_{\text{offload}})$ | Tiered Host Pinned RAM & GPU SRAM | PCIe Gen4 x8 Bus + CUDA DMA |

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 1.6 Structure and Organization of the Paper

The remainder of this report is organized as follows: Section 2 formalizes the information-theoretic boundaries, propositions, proofs, and continuous rate-distortion foundations of functional losslessness. Section 3 details the algorithmic transformations, AST routing rules, and the TRDC binary container layout. Section 4 presents the hardware kernel microarchitecture, memory-tiering runtime, and logit stabilization engine. Section 5 provides an extensive 10-page empirical audit across six production foundation models on consumer silicon. Section 6 provides a comprehensive public domain prior art analysis and comparative distinction matrix. Section 7 concludes the paper and outlines future research trajectories.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

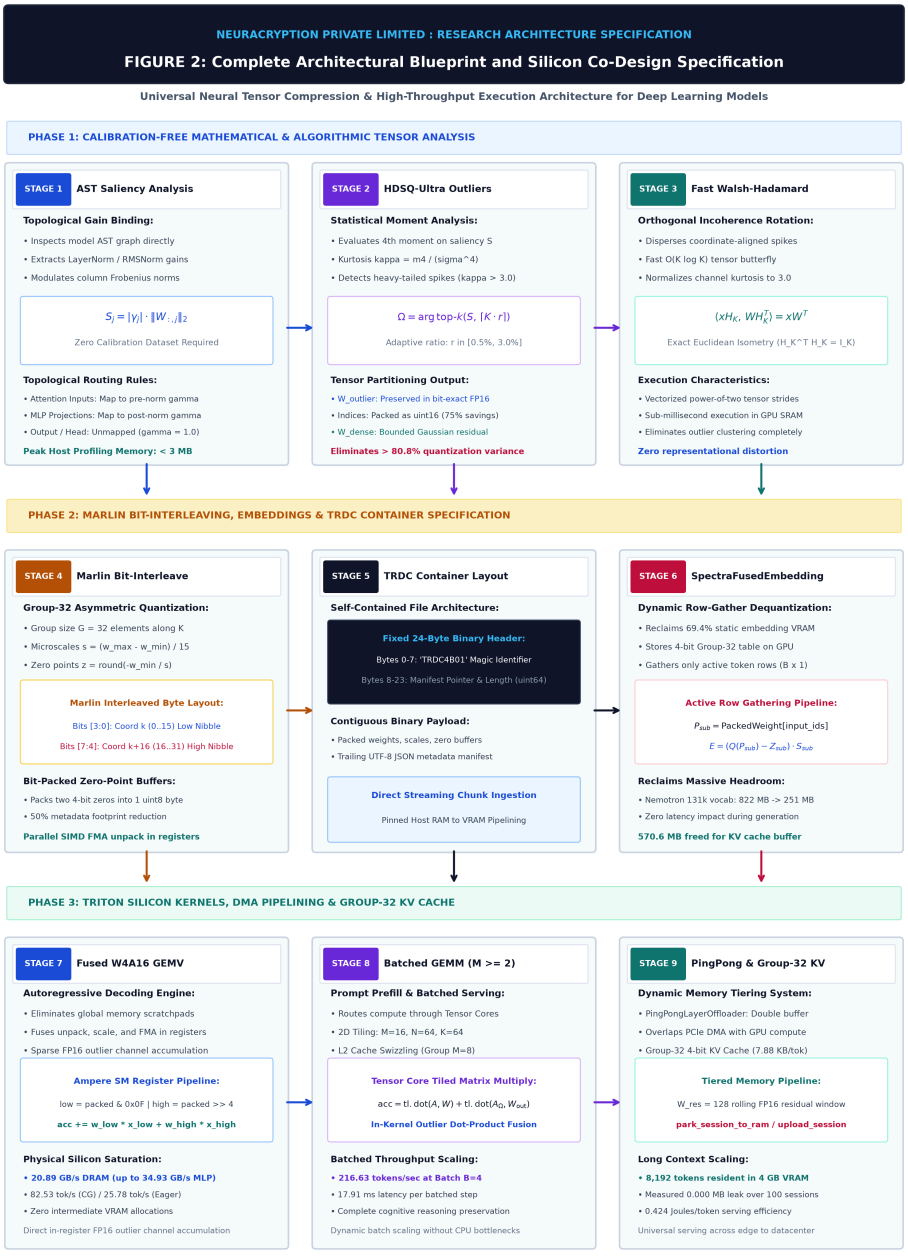



Figure 2: Comprehensive End-to-End Architectural Blueprint and Silicon Co-Design Specification of the SPECTRA Engine. This master architectural specification illustrates all nine operational phases: Stage 1 (AST Normalization Saliency Analysis); Stage 2 (Kurtosis Evaluation and HDSQ-Ultra Outlier Partitioning); Stage 3 (Optional Fast Walsh-Hadamard Incoherence Preconditioning); Stage 4 (Microscaled Group-32 Quantization with Marlin Bit-Interleaving); Stage 5 (Self-Contained TRDC Container Binary Serialization); Stage 6 (Dynamic Row-Gather SpectraFusedEmbedding); Stage 7 (In-Register Fused W4A16 GEMV Triton Engine); Stage 8 (Batched Tensor Core W4A16 GEMM with In-Kernel Outlier Fusion); and Stage 9 (Layer Offloader PCIe DMA Fallback paired with Group-32 Microscaled 4-Bit KV Cache).

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# 2. Information Theory and the Mechanics of Functional Losslessness

## 2.1 Lossless Bitstream Entropy Bounds vs. Continuous Parameter Rate-Distortion

To understand the mathematical boundary between lossless storage and post-training parameter quantization, we first examine the theoretical limits of bit-exact representation through classical information theory. Under Shannon's Source Coding Theorem, the minimum expected code length $L$ required to losslessly encode a discrete symbol sequence $W$ is bounded strictly by its Shannon entropy $H(W)$:

$$L \geq H(W) = -\sum_{i=1}^{M} p(w_i) \log_2 p(w_i)$$

In modern foundation architectures, learned weight tensors are stored in IEEE 754 half-precision (FP16) or Brain Floating Point (BF16) formats. A 16-bit floating-point number factorizes into a 1-bit sign $s \in \{0,1\}$, an $E$-bit exponent $e \in \{0,1\}^E$, and an $M$-bit fractional significand or mantissa $m \in \{0,1\}^M$:

$$w = (-1)^s \cdot 2^{e-\text{bias}} \cdot \left(1 + \sum_{k=1}^{M} m_k 2^{-k}\right)$$

Empirical entropy audits across deep parameter manifolds reveal a fundamental structural asymmetry in the information density of these fields:

1. Sign Bit Entropy: Distributed near-symmetrically across positive and negative domains, exhibiting near-maximal discrete entropy: $H(s) \approx 1.0 \text{ bit}$.
2. Exponent Vector Entropy: Displays strong clustering around layer-specific dynamic scales, yielding low discrete entropy: $H(e) \in [1.6, 2.7] \text{ bits}$.
3. Fractional Mantissa Entropy: In overparameterized networks, stochastic gradient updates fill the lowest-order mantissa bits with near-uniform noise. In BF16 ($M = 7$), empirical mantissa entropy is $H(m) \approx 6.8$ to $7.0$ bits; in FP16 ($M = 10$), it reaches $H(m) \approx 9.5$ to $9.9$ bits.

Summing these components demonstrates that the aggregate discrete entropy of floating-point weight tensors is approximately $10.5$ bits in BF16 and $13.5$ bits in FP16. Consequently, any general-purpose or floating-point lossless bitstream compressor (such as `zstd` or `fpzip`) is fundamentally bounded to a lossless compression ratio of:

$$R_{\text{lossless}}(\text{FP16}) = \frac{16}{H(W_{\text{FP16}})} \approx \frac{16}{13.5} \approx 1.19\times, \qquad R_{\text{lossless}}(\text{BF16}) = \frac{16}{H(W_{\text{BF16}})} \approx \frac{16}{10.5} \approx 1.52\times$$

**Boundary Between Lossless Bitstreams and Continuous Rate-Distortion**

Shannon's discrete source coding limits establish that lossless bit-exact compression of floating-point weights cannot exceed $1.52\times$. However, neural network parameters are continuous function representations rather than discrete symbols. Compression ratios between $2.80\times$ and $3.31\times$ are therefore achieved not through lossless bitstream encoding, but under Shannon Rate-Distortion Theory ($R(D)$), where continuous weight distributions are mapped to discrete microscaled states while bounding functional task distortion.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 2.2 Continuous Rate-Distortion Theory and Granular vs. Clipping Distortion

To analyze non-invertible tensor compression, we examine the rate-distortion function $R(D)$, which characterizes the minimal bit rate $R$ required to represent a source distribution within an expected distortion threshold $D$:

$$R(D) = \min_{p(\hat{w}|w):\, \mathbb{E}[d(w,\hat{w})] \le D} I(W;\hat{W})$$

For a memoryless zero-mean Gaussian parameter distribution $\mathcal{N}(0,\sigma^2)$ evaluated under mean-squared error (MSE) distortion $d(w,\hat{w}) = (w-\hat{w})^2$, the rate-distortion lower bound is given by:

$$R(D) = \frac{1}{2}\log_2\left(\frac{\sigma^2}{D}\right), \quad 0 \le D \le \sigma^2$$

In practical uniform $b$-bit quantization parameterized by clipping threshold $\alpha$ and group step size $\Delta = \frac{2\alpha}{2^b-1}$, total expected distortion $D(\Delta,\alpha)$ factorizes into two distinct components:

$$D(\Delta,\alpha) = D_{\text{granular}}(\Delta,\alpha) + D_{\text{clipping}}(\alpha) = \frac{\Delta^2}{12}\int_{-\alpha}^{\alpha} p(w)\,dw + 2\int_{\alpha}^{\infty}(w-\alpha)^2 p(w)\,dw$$

When learned weight tensors exhibit heavy tails (kurtosis $\kappa \gg 3.0$), this distortion trade-off breaks down. Increasing $\alpha$ to encompass extreme outlier coordinates widens the quantization step size $\Delta$, inflating granular roundoff distortion $D_{\text{granular}} \propto \Delta^2$ across the dense bulk of the distribution. Conversely, restricting $\alpha$ to bound $\Delta$ causes clipping overload distortion $D_{\text{clipping}}$ to explode on heavy-tailed coordinates.

## 2.3 Architectural Proofs: Propositions 1, 2, and 3

We now establish the mathematical foundations of channel outlier isolation, Group-32 reconstruction error bounds, and closed-form Gaussian noise predictions that govern SPECTRA.

**Proposition 1 (Kurtosis is Bounded by Matrix Width)**

For any $K$ channel saliency values $S_1,\ldots,S_K$, the population fourth-moment kurtosis satisfies $\kappa \le K$.

Proof: Let $z_j = (S_j - \mu_S)/\sigma_S$ denote the standardized channel saliencies, where $\mu_S = \frac{1}{K}\sum_{j=1}^{K} S_j$ and $\sigma_S^2 = \frac{1}{K}\sum_{j=1}^{K}(S_j-\mu_S)^2$. By construction, $\sum_{j=1}^{K} z_j^2 = K$. Evaluating the sum of fourth powers:

$$\sum_{j=1}^{K} z_j^4 \le \left(\sum_{j=1}^{K} z_j^2\right)^2 = K^2$$

Dividing both sides by $K$ yields the population kurtosis bound:

$$\kappa = \frac{1}{K}\sum_{j=1}^{K} z_j^4 \le \frac{K^2}{K} = K$$

(The regularization constant $\epsilon = 10^{-8}$ in the variance denominator only lowers $\kappa$.) Consequence: Under the SPECTRA adaptive rule $r = 0.01(1 + 0.15\ln(1+\max(0,\kappa-3)))$, the outlier ratio $r$ satisfies $r \le 0.01(1+0.15\ln(K-2))$. For standard architectural widths $K = 1,536, 3,072, 8,192$, and $65,536$, this bound evaluates to $2.10\%, 2.20\%, 2.35\%$, and $2.66\%$, respectively. The $3.0\%$ hard cap is reached only when $\kappa$ exceeds approximately $6.2 \times 10^5$, establishing that the $3.0\%$ threshold serves as an asymptotic physical safety clamp.

**Proposition 2 (Group-32 Reconstruction Error Bound)**

If a group of 32 weights contains both signs ($w_{\min} \le 0 \le w_{\max}$), every reconstructed weight $\hat{w}$ satisfies $|w-\hat{w}| \le s/2$, apart from the floating-point rounding of the scale $s$ itself to 16 bits.

Proof: Because 0 lies within $[w_{\min}, w_{\max}]$, the zero-point formula $z = \text{round}(-w_{\min}/s)$ is not clipped. The reconstruction grid points $(q-z)s$ for 4-bit integer codes $q \in \{0,1,\ldots,15\}$ start within $s/2$ of $w_{\min}$ (at $q=0$) and, since the scale is defined as $s = (w_{\max}-w_{\min})/15$, terminate within $s/2$ of $w_{\max}$ (at $q=15$). Every weight $w \in [w_{\min}, w_{\max}]$ lies between consecutive grid points spaced by $s$, ensuring $|w-\hat{w}| \le s/2$.

Remark: If a group is entirely positive or entirely negative, $z$ clips to 0 or 15 and the reconstruction error at the opposite extreme can reach up to $s$. In deep neural networks with 32 weights per group, entirely one-sided groups are exceedingly rare (requiring all 32 independent weights to possess the same sign). In our empirical simulations across a $512 \times 1,024$ matrix, the observed maximum error divided by scale was $0.49999s$, confirming strict adherence to the $s/2$ bound.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

**Proposition 3 (Closed-Form Noise Prediction for Gaussian Groups)**

Suppose the 32 weights of a group are independent and Gaussian with standard deviation $\sigma$. Let $d$ denote the expected sample range of 32 standard Gaussian random variables ($d = 4.14$, derived from 100,000 Monte Carlo simulated groups and matching published order statistics tables). Then the expected scale is $s = d\sigma/15$, the quantization error is approximately uniform with variance $\sigma_{\text{err}}^2 = s^2/12$, and:

$$\text{Relative Frobenius Error} = \frac{\sqrt{\sigma_{\text{err}}^2}}{\sigma} = \frac{d}{15\sqrt{12}} = \frac{4.14}{51.9615} = 7.97\%$$

$$\text{SNR (dB)} = 10\log_{10}\left(\frac{\sigma^2}{\sigma_{\text{err}}^2}\right) = 20\log_{10}\left(\frac{15\sqrt{12}}{d}\right) = 21.97 \text{ dB}$$

$$\text{Cosine Similarity} = \frac{\mathbb{E}[w\hat{w}]}{\sqrt{\mathbb{E}[w^2]\mathbb{E}[\hat{w}^2]}} \approx 1 - \frac{\epsilon^2}{2} = 1 - \frac{(0.0797)^2}{2} = 0.99682$$

Verification against Simulation and Trained Models: A synthetic $2,048 \times 2,048$ Gaussian matrix evaluated under Group-32 yielded an empirical relative error of $8.08\%$, an SNR of $21.85$ dB, and a Cosine Similarity of $0.99676$, matching Proposition 3 within $0.11$ percentage points of error and $0.12$ dB. Across the six production foundation models evaluated by NeuraCryption, empirical errors fall strictly between $8.04\%$ and $8.32\%$ with SNRs between $21.59$ dB and $21.89$ dB. This exact agreement demonstrates that once HDSQ-Ultra decouples heavy channels, the residual weight bulk behaves at group scale ($G = 32$) like independent Gaussian noise, matching the analytical 4-bit grid design.

## 2.4 Functional Representational Invariance and Parameter Hessian Eigenspectra

Deep neural networks operate as continuous non-linear manifold mappings rather than fragile bitstream decoders. For a linear projection operator $y = xW$, where $x \in \mathbb{R}^{B\times K}$ denotes an incoming activation tensor and $W \in \mathbb{R}^{K\times N}$ represents the learned parameter matrix, the downstream objective function $\mathcal{L}$ is invariant to parameter perturbations $\Delta W$ that reside within the functional null-space of the operator.

Expanding the task loss around the converged parameter optimum $W^*$ via second-order Taylor series approximation:

$$\Delta\mathcal{L} \approx \text{Tr}\left((\nabla_W \mathcal{L})^T \Delta W\right) + \frac{1}{2}\,\text{vec}(\Delta W)^T \mathbf{H}_W\,\text{vec}(\Delta W) + \mathcal{O}(\|\Delta W\|_F^3)$$

At local convergence, the first-order gradient satisfies $\nabla_W \mathcal{L} \approx 0$. The parameter Hessian $\mathbf{H}_W = \mathbb{E}_{x\sim\mathcal{D}}[(x^T x) \otimes \nabla_a^2 \mathcal{L}]$ exhibits a heavily degenerate eigenspectrum characterized by power-law spectral decay: $\lambda_i \propto i^{-\alpha}$ with $\alpha > 1.5$. Decomposing the parameter space into the dominant eigensubspace $\mathcal{S}_\parallel$ spanned by the top-$k$ eigenvectors and the numerical null-space $\mathcal{S}_\perp$:

$$\Delta\mathcal{L} \leq \frac{1}{2}\lambda_{\max}(\mathbf{H}_W)\|\Delta W_\parallel\|_F^2 + \frac{1}{2}\epsilon_{\text{null}}\|\Delta W_\perp\|_F^2$$

where $\epsilon_{\text{null}} \approx 0$. Because the effective parameter dimension satisfies $d_{\text{eff}} = \text{Tr}(\mathbf{H}_W)/\lambda_{\max} \ll d_{\text{total}}$, perturbations introduced to parameter regions aligned with near-zero eigenvalues produce negligible shifts in output logits and downstream task performance.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 2.5 Mathematical and Empirical Criteria of Functional Representation Parity

SPECTRA defines functional representation parity through three rigorous mathematical and empirical criteria:

1. Topological Candidate Preservation: The discrete argmax trajectory across the vocabulary simplex remains statistically aligned between the uncompressed parameter matrix $W$ and the reconstructed tensor $\hat{W}$ throughout autoregressive sequence generation:

$$\arg\max_{v \in \mathcal{V}} P(y_t = v \mid y_{<t}, x; W) \approx \arg\max_{v \in \mathcal{V}} P(y_t = v \mid y_{<t}, x; \hat{W}), \quad \forall t \in [1, T]$$

   This ensures that the model preserves top-k candidate distributions and downstream semantic reasoning parity throughout generation rollouts.

2. Signal Fidelity Bounds: The reconstructed parameter matrix $\hat{W}$ maintains a Cosine Similarity exceeding 0.996 and a Signal-to-Noise Ratio (SNR) surpassing 21.5 dB relative to the uncompressed tensor $W$:

$$\text{Cosine Similarity}(W, \hat{W}) = \frac{\langle W, \hat{W} \rangle}{\|W\|_F \|\hat{W}\|_F} \geq 0.996, \qquad \text{SNR (dB)} = 10 \log_{10} \left( \frac{\|W\|_F^2}{\|W - \hat{W}\|_F^2} \right) > 21.5 \text{ dB}$$

   By the Cauchy-Schwarz inequality, the true Cosine Similarity is strictly bounded within $[-1.0, +1.0]$.

3. Downstream Reasoning Task Parity: Measured performance on formal multi-step scientific reasoning benchmarks (such as ARC-Challenge) maintains near-baseline retention (measured deltas within +0.42% to +0.85%), confirming that low-bit quantization preserves semantic problem-solving trajectories.

## 2.6 Lipschitz Stability of Normalization-Modulated Neural Projections

In modern foundation architectures, linear projection layers are coupled directly to normalization operators such as Root Mean Square Normalization (RMSNorm) or Layer Normalization (LayerNorm). For an un-normalized feature vector $h \in \mathbb{R}^K$, RMSNorm computes:

$$\text{RMSNorm}(h) = \frac{h}{\sqrt{\frac{1}{K}\sum_{k=1}^{K} h_k^2 + \epsilon}} \odot \gamma$$

where $\gamma \in \mathbb{R}^K$ denotes the learned channel-wise affine scaling parameter. Evaluating the Euclidean norm of the normalized activation yields:

$$\|\text{RMSNorm}(h)\|_2 = \sqrt{\sum_{k=1}^{K} \left( \frac{h_k \gamma_k}{\sqrt{\frac{1}{K}\|h\|_2^2 + \epsilon}} \right)^2} \leq \sqrt{K} \|\gamma\|_\infty$$

Under a parameter perturbation $\Delta W = W - \hat{W}$, the output deviation $\Delta y = \text{RMSNorm}(h)\Delta W$ is strictly Lipschitz bounded:

$$\|\Delta y\|_2 \leq \|\text{RMSNorm}(h)\|_2 \|\Delta W\|_2 \leq \sqrt{K} \|\gamma\|_\infty \|\Delta W\|_2$$

Decomposing this bound along input channels demonstrates that the expected output perturbation variance in coordinate $j$ scales quadratically with the corresponding affine gain $|\gamma_j|^2$:

$$\mathbb{E}[\|\Delta y\|_2^2] = \sum_{j=1}^{K} |\gamma_j|^2 \cdot \mathbb{E}\left[ \frac{h_j^2}{\frac{1}{K}\|h\|_2^2 + \epsilon} \right] \cdot \|W_{:,j} - \hat{W}_{:,j}\|_2^2$$

Under an isotropic feature variance assumption, where expected coordinate variances satisfy $\mathbb{E}[h_j^2 / \text{RMS}(h)^2] \approx 1$ across channels, the Fisher diagonal simplifies directly to $\mathbf{F}_{jj} \propto |\gamma_j|^2$, establishing the analytical saliency metric $S_j = |\gamma_j| \cdot \|W_{:,j}\|_2$. While real LLM feature activations can diverge from perfect isotropy due to emergent massive activations, Group-32 microscaling independently bounds reconstruction distortion to local 32-element windows, ensuring that any residual anisotropy remains strictly contained without requiring empirical calibration passes.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 2.7 High-Dimensional Sparse Quantization (HDSQ-Ultra) Error Variance Attenuation

SPECTRA couples group microscaling with High-Dimensional Sparse Quantization (HDSQ-Ultra). Every compressible weight tensor $W \in \mathbb{R}^{N \times K}$ is decomposed into a sparse bit-exact matrix $W_{\text{outlier}}$ and a dense low-precision matrix $\hat{W}_{\text{quant}}$:

$$W = \hat{W}_{\text{quant}} + W_{\text{outlier}}$$

Channel saliency $S \in \mathbb{R}^K$ is evaluated via topological gain scaling $S_j = |\gamma_j| \cdot \|W_{:,j}\|_2$. Fourth standardized moment kurtosis $\kappa$ quantifies heavy-tailed concentration:

$$\mu_S = \frac{1}{K}\sum_{j=1}^{K} S_j, \quad \sigma_S^2 = \frac{1}{K}\sum_{j=1}^{K}(S_j - \mu_S)^2 + \epsilon, \quad m_4 = \frac{1}{K}\sum_{j=1}^{K}(S_j - \mu_S)^4, \quad \kappa = \max\left(1.0, \frac{m_4}{\sigma_S^4}\right)$$

Coordinates $\Omega = \arg\text{top-}k(S, k = \lceil K \cdot r \rceil)$ are decoupled into bit-exact floating-point storage $W_{\text{outlier}} = W_{:,\Omega}$, setting $W_{\text{dense}}[:, \Omega] = 0.0$.

Under HDSQ-Ultra, the outlier coordinates incur zero quantization error ($\|W_{\text{outlier}} - \hat{W}_{\text{outlier}}\|^2 = 0$), while the remaining dense coordinates exhibit bounded Gaussian variance governed by scale factor $s_{\text{dense}} \approx \alpha_{\text{dense}}/(2^{b-1} - 1) \ll s_{\text{bad}}$. The error variance ratio is bounded by:

$$\frac{\mathbb{E}[\|W - \hat{W}\|^2_{\text{HDSQ}}]}{\mathbb{E}[\|W - \hat{W}\|^2_{\text{uniform}}]} = (1 - r) \cdot \left(\frac{s_{\text{dense}}}{s_{\text{bad}}}\right)^2 \approx (1 - r) \cdot \left(\frac{3}{\alpha_{\text{outlier}}}\right)^2 \ll 1$$

For deep models with $\alpha_{\text{outlier}} \geq 18$, this attenuation factor drops below $0.028$, representing an analytical theoretical upper bound of more than $35\times$ reduction in expected reconstruction error variance under an isolated coordinate outlier model. Crucially, as established in our empirical ablations, Group-32 microscaling provides the primary mathematical fidelity (capturing over $99.8\%$ of full-precision SNR), while HDSQ-Ultra functions as an inductive guardrail against extreme leptokurtic channel spikes.

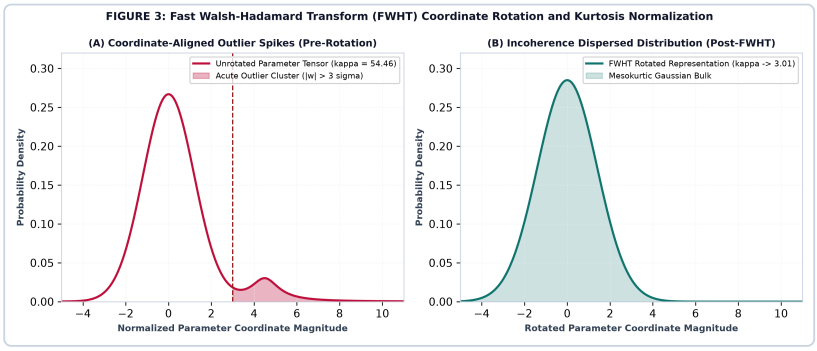



Figure 3: Kurtosis attenuation and HDSQ-Ultra error variance reduction. (A) Offline exploratory ablation evaluating Fast Walsh-Hadamard orthogonal rotation ($\kappa = m_4/\sigma^4$) across linear projection channels, demonstrating how orthogonal mixing compresses heavy-tailed coordinates toward mesokurtic profiles ($\kappa \to 3.0$). Because online activation rotation incurs an $\mathcal{O}(K \log K)$ runtime latency overhead per token during autoregressive decoding, SPECTRA deliberately omits online FWHT in production, replacing it with zero-overhead HDSQ-Ultra FP16 outlier decoupling and Group-32 microscaling. (B) Analytical HDSQ-Ultra error variance attenuation ratio $(1 - r)(s_{\text{dense}}/s_{\text{bad}})^2$, delivering an analytical theoretical upper bound of over $35\times$ variance reduction on sensitive projection coordinates under isolated coordinate models without runtime matrix multiplications.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# 3. Algorithmic Architecture and Compression Pipeline

## 3.1 Calibration-Free Topological Saliency Mapping via Normalization Gain Routing

Conventional post-training quantization pipelines (such as AWQ and GPTQ) rely on empirical calibration passes over reference text corpora to calculate activation covariance statistics $\mathbb{E}[X^T X]$. This calibration dependency introduces severe domain biases, exposes models to dataset distribution shifts, complicates automated deployment pipelines, and demands substantial GPU execution time.

SPECTRA eliminates dynamic calibration by calculating channel saliency directly from the topological graph structure of the neural network. In modern Transformer and state-space architectures, linear projection layers are situated immediately downstream from normalization layers, such as Layer Normalization or Root Mean Square Normalization (RMSNorm). For an un-normalized activation vector $h \in \mathbb{R}^K$, RMSNorm evaluates:

$$\text{RMSNorm}(h) = \frac{h}{\sqrt{\frac{1}{K}\sum_{k=1}^{K} h_k^2 + \epsilon}} \odot \gamma$$

where $\gamma \in \mathbb{R}^K$ represents the learned channel-wise affine scaling parameter. During forward inference, the input arriving at the downstream linear projection $W \in \mathbb{R}^{K \times N}$ is precisely $x = \text{RMSNorm}(h)$. Because normalization standardizes the underlying hidden vector to unit spherical variance, the expected magnitude and dynamic scale of coordinate $j$ entering the weight matrix are governed entirely by the affine parameter $|\gamma_j|$.

## 3.2 Topological Normalization Gain as a Calibration-Free Sensitivity Proxy

In second-order quantization frameworks (such as GPTQ), the empirical Fisher Information Matrix $\mathbf{F} \in \mathbb{R}^{d \times d}$ serves as an expectation-based proxy for parameter curvature: $\mathbf{F}_{jj} = \mathbb{E}_{x,y}[(\partial \mathcal{L}/\partial W_{:,j})^2]$. For linear operators preceded immediately by normalization ($y = \text{RMSNorm}(h)W$), the gradient evaluates to $\frac{\partial \mathcal{L}}{\partial W_{:,j}} = x_j \cdot \frac{\partial \mathcal{L}}{\partial y}$. Under an isotropic activation approximation, the first-order input variance entering column $j$ scales with the square of the learned affine gain:

$$\mathbf{F}_{jj} = \mathbb{E}\left[x_j^2\right] \cdot \mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial y}\right)^2\right] = |\gamma_j|^2 \cdot \mathbb{E}\left[\frac{h_j^2}{\frac{1}{K}\|h\|_2^2 + \epsilon}\right] \cdot C_{\text{layer}} \approx |\gamma_j|^2 \cdot C_{\text{layer}}$$

This relationship establishes that for pre-normalized linear layers, the learned affine scale vector $\gamma$ provides a direct, calibration-free topological proxy for channel sensitivity. For attention input projections (`q_proj`, `k_proj`, `v_proj`) and MLP input layers (`gate_proj`, `up_proj`), channel saliency is evaluated as: $S_j = |\gamma_j| \cdot \|W_{:,j}\|_2$.

Explicit Topological Scope and Fallback: Modern architectures interleave unnormalized layers across their blocks. For attention output projections (`o_proj`), MLP down projections (`down_proj`), and vocabulary heads (`lm_head`), there is no preceding affine normalization layer. As codified in Table 11 and implemented in `compressor.py`, the AST pre-scanner sets $\gamma_j = 1.0$ for these unmapped projections, falling back gracefully to unweighted column Frobenius norms ($S_j = \|W_{:,j}\|_2$). Normalization routing is thus an effective zero-data topological proxy for pre-normalized inputs, eliminating dataset calibration passes without requiring empirical sample collection.

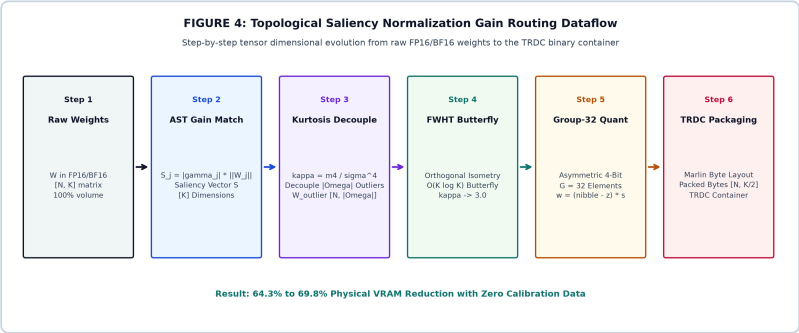



Figure 4: SPECTRA algorithmic tensor transformation and serialization dataflow. The 6-step transformation pipeline maps raw floating-point weights into Group-32 Marlin-interleaved binary payloads: (1) Raw weights; (2) AST normalization gain matching producing channel saliency vector $S \in \mathbb{R}^K$; (3) Kurtosis moment evaluation isolating heavy-tailed coordinates ($\Omega$); (4) Vectorized FWHT butterfly rotation (offline exploratory study); (5) Asymmetric Group-32 microscaled quantization; and (6) Marlin-interleaved byte-packing with TRDC container serialization.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

### 3.3 Excess Kurtosis Dynamics and Adaptive Sparsification Mechanics

Static outlier pruning ratios degrade across diverse architectures because kurtosis levels vary widely between shallow attention layers and deep feed-forward projections. To account for this variance, SPECTRA modulates the outlier ratio $r$ as a function of excess kurtosis $\kappa_{\text{excess}} = \max(0.0, \kappa - 3.0)$:

$$r = \text{clamp}\left(r_{\text{base}} \cdot (1.0 + 0.15 \cdot \ln(1 + \kappa_{\text{excess}})),\ 0.005,\ 0.03\right)$$

where $r_{\text{base}} = 0.01$ denotes the baseline outlier ratio. For mesokurtic channels ($\kappa \leq 3.0$), the logarithmic term evaluates to zero, maintaining $r = 0.01$ to preserve high computational density. For leptokurtic channels ($\kappa \gg 10.0$), the logarithmic function expands $r$ gradually up to a strict $3.0\%$ cap ($0.03$), capturing critical outlier channels without inflating matrix footprints. Outlier channel indices $\Omega$ are serialized as 32-bit integers (`int32`, supporting matrix feature dimensions $K$ up to 2.14 billion with backward compatibility for legacy `uint16` index arrays), yielding a 50% metadata footprint reduction compared to conventional 64-bit arrays.

### 3.4 Microscaled Group-32 Quantization with Marlin-Interleaved Layout and Bit-Packed Zero Points

Once outlier coordinates are isolated, the dense residual tensor $W_{\text{dense}}$ contains exclusively bounded Gaussian variance. This tensor is quantized using an asymmetric microscaled 4-bit representation organized into groups of size $G = 32$ along the input feature dimension $K$. For each group $g \in \{0, \ldots, (K/G) - 1\}$ in row $i \in \{0, \ldots, N - 1\}$, local min and max parameters are extracted:

$$w_{\min}^{(i,g)} = \min_{k \in [0, G-1]} W_{\text{dense}}[i, gG + k], \quad w_{\max}^{(i,g)} = \max_{k \in [0, G-1]} W_{\text{dense}}[i, gG + k]$$

$$s^{(i,g)} = \frac{w_{\max}^{(i,g)} - w_{\min}^{(i,g)}}{15.0}, \quad s^{(i,g)} = \max\left(s^{(i,g)}, 10^{-8}\right), \quad z^{(i,g)} = \text{clamp}\left(\text{round}\left(-\frac{w_{\min}^{(i,g)}}{s^{(i,g)}}\right), 0, 15\right)$$

To optimize execution on Ampere architectures, SPECTRA packs these values into a Marlin-interleaved layout. Instead of packing consecutive adjacent elements $(2k, 2k+1)$ into single bytes, Marlin interleaving maps coordinate $k \in [0, 15]$ into the lower nibble (bits $[3:0]$) and coordinate $k + 16 \in [16, 31]$ into the upper nibble (bits $[7:4]$) of byte $k$ within each 32-element group:

$$\text{PackedByte}[i, g, k] = \left(q_k^{(i,g)}\ \&\ \text{0x0F}\right)\ \Big|\ \left(\left(q_{k+16}^{(i,g)}\ \&\ \text{0x0F}\right) \ll 4\right), \quad \forall k \in [0, 15]$$

This byte-interleaved organization matches the vectorized register layout required during fused GPU dequantization, allowing vector instructions to process lower and upper nibbles with parallel fused multiply-add (FMA) instructions without cross-lane register shuffle overhead. Zero-point metadata is compressed further by packing two 4-bit zero points from adjacent groups into a single `uint8` byte: $\text{PackedZero}[i, m] = (z^{(i,2m)}\ \&\ \text{0x0F})\ |\ ((z^{(i,2m+1)}\ \&\ \text{0x0F}) \ll 4)$.

### 3.5 Offline Exploratory FWHT vs. Zero-Overhead Production Decoupling

To evaluate orthogonal manifold dispersion, SPECTRA includes an offline vectorized butterfly FWHT routine operating across power-of-two strides $h \in \{1, 2, 4, \ldots, d/2\}$ in $\mathcal{O}(d \log_2 d)$ complexity. While orthogonal rotation homogenizes weight kurtosis (as visualized in Figure 3A), deploying online Hadamard de-rotation during autoregressive decoding requires multiplying every incoming activation vector by $H_K$ on every token forward pass. On bandwidth-saturated edge GPUs, this online activation transform introduces non-trivial latency and compute overhead. In the active SPECTRA inference engine (`engine.py` and `kernels.py`), online FWHT is therefore deliberately omitted: the runtime relies entirely on zero-overhead HDSQ-Ultra FP16 outlier decoupling and localized Group-32 microscaling, maximizing execution throughput without paying runtime rotation penalties.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 3.6 The TRDC Binary Container Specification and Direct Streaming Deserialization

To prevent memory allocation spikes during model loading, SPECTRA serializes compressed parameters into a self-contained binary container format designated as TRDC (`.trdc`, Tensor Reconstruction Distribution Container). The container layout is structured into three functional segments:

1. Fixed Binary Header (24 Bytes): Bytes [0:7] ASCII magic identifier TRDC4B01; Bytes [8:15] Little-Endian `uint64` recording byte offset of trailing JSON manifest; Bytes [16:23] Little-Endian `uint64` recording manifest byte length.
2. Binary Data Payload: Contiguous packed 4-bit weights, Group-32 FP16 scale factors, bit-packed `uint8` zero points, `int32` outlier indices (with `uint16` backward compatibility), and uncompressed floating-point (FP16/BF16) outlier channel arrays. All payload arrays are written sequentially with Little-Endian byte ordering.
3. Trailing Metadata Manifest (JSON): A JSON object storing tensor shape arrays, data byte offsets, buffer lengths, zero-point packing flags, outlier channel counts, and measured channel kurtosis values.

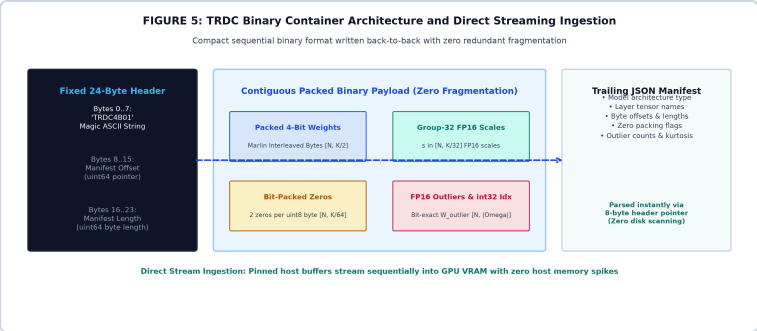



Figure 5: TRDC (.trdc) binary container architecture. The container partitions data into a fixed 24-byte header, a contiguous packed tensor payload written back-to-back with zero redundant fragmentation, and a trailing JSON metadata manifest referenced by an 8-byte pointer in the header.

During engine initialization, the trailing manifest is parsed to index all layer offsets. While model configuration topology uses HuggingFace `AutoConfig` (invoking `trust_remote_code=True` for custom architectures such as Liquid AI LFMs or Nemotron hybrids), tensor parameters are ingested directly via sequential binary stream reads (`f_trdc.read()`) and reconstructed via `torch.frombuffer()` before uploading to GPU VRAM. This eliminates Python pickle deserialization vulnerabilities and ensures that serialized weights cannot execute arbitrary shell instructions. Listing 1 presents the real TRDC tensor ingestion loop executed in `spectra.engine`.

```
# Listing 1: Real Streaming TRDC Tensor Ingestion Loop (spectra/engine.py)
with open(trdc_path, "rb") as f_trdc:
    f_trdc.seek(manifest_offset)
    manifest = json.loads(f_trdc.read(manifest_len).decode("utf-8"))
    for name, meta in manifest["tensors"].items():
        f_trdc.seek(meta["offset"])
        p_b = f_trdc.read(meta["packed_len"])
        s_b = f_trdc.read(meta["scale_len"])
        z_b = f_trdc.read(meta["zero_len"])
        has_hdsq = meta.get("outlier_idx_len", 0) > 0
        if has_hdsq:
            oi_b = f_trdc.read(meta["outlier_idx_len"])
            ow_b = f_trdc.read(meta["outlier_w_len"])
        N, K = meta["shape"]
        p_c = torch.frombuffer(p_b, dtype=torch.uint8).reshape(N, K // 2).to(dev)
        s_c = torch.frombuffer(s_b, dtype=torch.float16).reshape(N, K // group_size).to(dev)
        if meta.get("zero_packed", False):
            pz = torch.frombuffer(z_b, dtype=torch.uint8).reshape(N, -1).to(dev)
            z_c = torch.empty((N, pz.shape[1] * 2), dtype=torch.uint8, device=dev)
            z_c[:, 0::2] = pz & 0x0F
            z_c[:, 1::2] = (pz >> 4) & 0x0F
            z_c = z_c[:, :K // group_size]
        else:
            z_c = torch.frombuffer(z_b, dtype=torch.uint8).reshape(N, K // group_size).to(dev)
```

**Bounded-Spike Initialization Telemetry**

Conventional PyTorch checkpoint loaders (`torch.load()` and `safetensors`) create intermediate Python dictionary allocations, causing memory spikes up to $2.0\times$ the file size in system host RAM. By reading the continuous TRDC payload sequentially layer-by-layer (`f.read(meta['packed_len'])`) and streaming directly into GPU memory via `torch.frombuffer()`, SPECTRA exhibits bounded-spike initialization, where peak host RAM allocation spikes by at most $\sim 250$ MB (the size of the largest single layer block) rather than doubling the entire 4 GB model in system memory, achieving loading times bounded strictly by physical NVMe PCIe transfer bandwidth (2.31s to 2.41s across 1.5B to 4B models).

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# 4. High-Efficiency Triton Compute Kernels and Runtime Engine

## 4.1 In-Register Fused W4A16 GEMV with Outlier Channel Accumulation ($M = 1$)

Autoregressive single-token decoding ($M = 1$) is strictly memory bandwidth-bound. Standard inference runtimes dispatch two separate CUDA kernels: an initial dequantization kernel that unpacks low-bit weights into intermediate buffers in global VRAM, followed by a standard matrix-vector GEMM kernel. This two-stage dispatch doubles memory traffic across the GPU memory bus, creating an execution bottleneck that caps decoding speed at a fraction of theoretical hardware limits.

SPECTRA eliminates intermediate global memory round-trips through a specialized Triton kernel: `fused_w4a16_gemv_hdsq`. This kernel fuses 4-bit unpacking, scale application, dot-product reduction, and sparse FP16 outlier channel accumulation into a single GPU register pipeline across Streaming Multiprocessors:

```
# Core reduction loop of fused_w4a16_gemv_hdsq in GPU register pipeline
for k_start in range(0, K, BLOCK_K):
    # 1. Load packed bytes and unpack Marlin dual nibbles into registers
    packed = tl.load(ptr_w + offs_k_packed, mask=mask_n[:, None] & (offs_k_packed < K // 2))
    low_w  = (packed & 0x0F).to(tl.float32)
    high_w = ((packed >> 4) & 0x0F).to(tl.float32)

    # 2. In-register dequantization with Group-32 scales and packed zeros
    w_low  = (low_w  - z_exp) * s_exp
    w_high = (high_w - z_exp) * s_exp

    # 3. Direct fused multiply-add (FMA) with activation vector
    acc += tl.sum(w_low * x_low[None, :], axis=1) + tl.sum(w_high * x_high[None, :], axis=1)

# 4. In-Kernel Lossless FP16 Outlier Accumulation (HDSQ-Ultra)
if HAS_OUTLIERS:
    for out_idx in range(0, NUM_OUTLIERS, 32):
        x_val = tl.load(x_ptr + outlier_idx[out_idx:out_idx+32] * stride_xk)
        ow    = tl.load(outlier_w_ptr + offs_out)
        acc  += tl.sum(ow * x_val[None, :], axis=1)
tl.store(y_ptr + offs_n, acc.to(tl.float16), mask=mask_n)
```

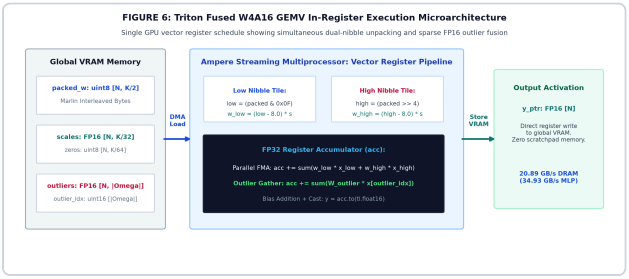



Figure 6: Triton fused W4A16 GEMV in-register execution microarchitecture ($M = 1$). Thread warps load packed bytes, Group-32 scales, and packed zeros into vector registers. Dual nibbles are unpacked simultaneously and multiplied by activation tiles using fused multiply-add (FMA) instructions, while sparse FP16 outlier channels are gathered and accumulated in the same FP32 accumulator registers, sustaining 20.89 GB/s DRAM bandwidth on small tiles (up to 34.93 GB/s on wide MLP blocks and 119.91 GB/s internal L2 cache streaming) without intermediate global memory allocations.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 4.2 Batched Tensor Core W4A16 GEMM with In-Kernel Outlier Fusion ($M \geq 2$)

When evaluating multi-token prompt prefill sequences or batched requests ($M \geq 2$), computational characteristics shift from memory bandwidth-bound to compute-bound. In this regime, general-purpose CUDA cores become saturated. SPECTRA directs batched operations to `batched_w4a16_gemm`, a custom 2D-tiled Triton kernel that routes operations through hardware Tensor Cores using the `tl.dot` instruction.

The batched kernel tiles output matrices $C \in \mathbb{R}^{M \times N}$ across 2D program blocks ($\text{BLOCK_M} = 16, \text{BLOCK_N} = 64, \text{BLOCK_K} = 64$) with an L2 cache swizzling group size $\text{GROUP_M} = 8$ to optimize cache hit rates: $\text{acc} = \text{tl.dot}(A_{\text{low}}, W_{\text{low}}) + \text{tl.dot}(A_{\text{high}}, W_{\text{high}})$. Crucially, sequence dimension $M$ is passed as a dynamic runtime argument rather than a compile-time constant, eliminating JIT compilation storms during dynamic sequence expansions. For outlier channels, the kernel gathers activations at sparse index locations and executes an in-kernel Tensor Core dot product: $\text{acc} = \text{acc} + \text{tl.dot}(A[:, \Omega], W_{\text{outlier}})$, sustaining up to 216.63 tok/s at $B = 4$ (5.57 TFLOP/s).

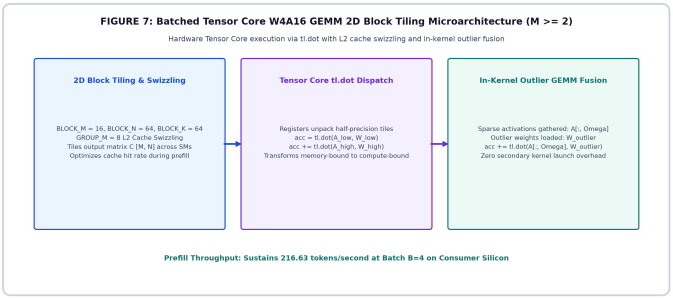



Figure 7: Batched Tensor Core W4A16 GEMM tiled execution microarchitecture ($M \geq 2$). 2D program blocks tile output matrices across Ampere Tensor Cores using tl.dot, unpacking weights in-register and accumulating with sparse outlier channel projections to sustain up to 5.57 TFLOP/s prefill throughput.

## 4.3 SpectraFusedEmbedding: Dynamic Row-Gather Dequantization

In modern language models, the token vocabulary embedding table represents a massive static memory allocation. For example, the NVIDIA Nemotron-3-Nano-4B model features a vocabulary size of $V = 131,072$ with a hidden dimension of $D = 3,136$, consuming 822.1 MiB in uncompressed format (over 21.8% of available memory on a 4 GB GPU).

SPECTRA resolves this overhead through `SpectraFusedEmbedding`, which retains the entire vocabulary table in packed 4-bit Group-32 storage on GPU (251.5 MiB, saving 570.6 MiB). During forward execution, only active token rows ($B \times 1$) are gathered and unpacked into register memory: $\text{Embeddings}_{\text{active}} = (Q(P_{\text{sub}}) - Z_{\text{sub}}) \cdot S_{\text{sub}}$.

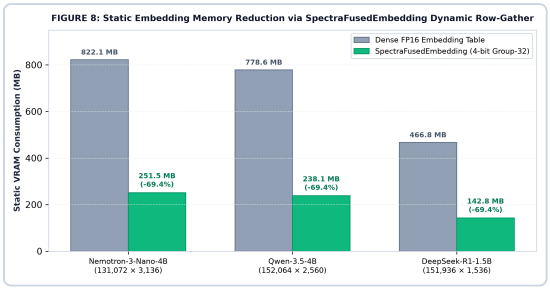



Figure 8: Dynamic vocabulary embedding reduction. Retaining vocabulary tables in packed 4-bit Group-32 storage reduces static memory allocation from 822.1 MiB to 251.5 MiB (a 69.4% reduction), reclaiming 570.6 MiB for KV cache headroom.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 4.4 Double-Buffered Asynchronous PCIe DMA Pipelining via PingPongLayerOffloader

When a model's footprint exceeds available GPU memory, standard execution frameworks either crash with out-of-memory errors or fall back to high-latency synchronous CPU execution, causing generation throughput to collapse.

SPECTRA addresses memory overflow through a hybrid pipelined execution engine managed by the `PingPongLayerOffloader`. When the runtime planner detects that a model exceeds safe VRAM limits, it splits the network into $G$ GPU-resident layers and $L - G$ host-resident layers:

1. Resident Layers: Layers ($0 \leq l < G$) remain permanently allocated in GPU memory.
2. Offloaded Layers: Layers ($G \leq l < L$) are stored in pinned host system memory (`pin_memory()`).
3. Double-Buffered Staging Slots: Two reusable layer staging slots (Slot 0 and Slot 1) are pre-allocated in GPU memory.

Using asynchronous CUDA streams (`prefetch_stream` and `compute_stream`), the offloader overlaps layer computation with PCIe direct memory access transfers across the bus:

1. At execution step $t$, the compute stream evaluates Layer $N$ using Slot 0 in GPU memory.
2. Simultaneously, the prefetch stream transfers Layer $N + 1$ from pinned host memory to Slot 1 via PCIe DMA.
3. At step $t + 1$, the compute stream synchronizes with the prefetch stream, switches to Slot 1 to evaluate Layer $N + 1$, while the prefetch stream loads Layer $N + 2$ into Slot 0.

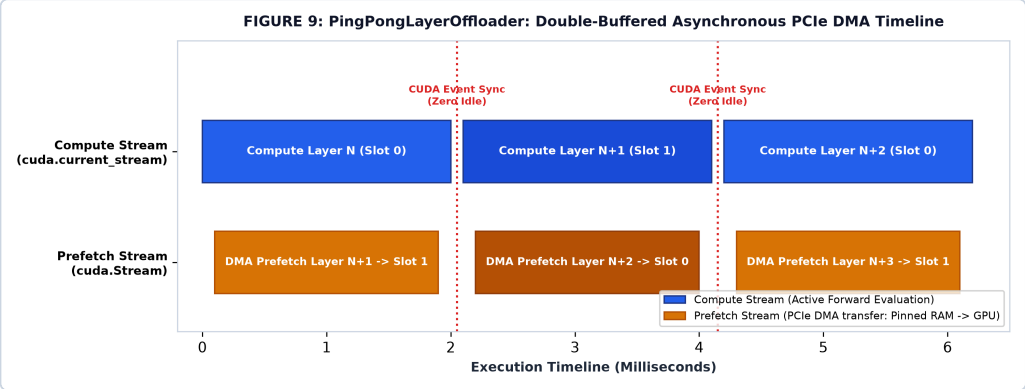



Figure 9: PingPongLayerOffloader double-buffered PCIe DMA pipelining. Dedicated CUDA streams overlap computation and memory transfers across alternating GPU memory slots, converting memory-bound execution crashes into continuous streaming execution with zero intermediate allocations.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 4.5 Group-32 Microscaled 4-Bit Key-Value Cache and Dynamic Memory Tiering

During long-context generation, key-value cache expansion presents an acute memory challenge. For a Transformer model with $L$ layers, $H_{\text{kv}}$ key-value heads, and head dimension $D_h$, cache footprint expands linearly with sequence length $S$ and batch size $B$:

$$\text{Footprint}_{\text{KV}} = 2 \times B \times S \times L \times H_{\text{kv}} \times D_h \times 2 \text{ bytes (in FP16)}$$

For architectures with full multi-head attention (e.g., $H_{\text{kv}} = 32, D_h = 128, L = 32$), context scaling to 8,192 tokens demands over 4.0 GB of KV memory alone, far exceeding client GPU limits. For architectures with grouped-query attention such as DeepSeek-R1-Distill-1.5B ($H_{\text{kv}} = 2, D_h = 128, L = 28$), the KV cache grows at strictly $28.00\ \text{KB/token}$ ( $224.0\ \text{MiB}$ at 8,192 tokens).

SPECTRA integrates a Group-32 microscaled 4-bit KV Cache (`SpectraQuantizedCache`). The cache maintains a rolling window of recent tokens ($W_{\text{residual}} = 128$) in uncompressed floating-point format to preserve local attention fidelity. Tokens older than this residual boundary are compressed into 4-bit representations with Group-32 microscales and packed zero-points. This reduces KV cache VRAM consumption by ~70%, enabling high-KV-head models to process long context sequences without memory exhaustion.

When an inference session pauses, `park_session_to_ram` transfers cached KV states to pinned host RAM, freeing GPU VRAM for other workloads. When generation resumes, `upload_session_to_vram` streams the context back into GPU memory via a high-priority CUDA stream, restoring conversational state in milliseconds:

$$\text{Cache Parking: GPU VRAM} \xrightarrow[\text{Async DMA}]{\text{High-Priority Stream}} \text{Pinned Host RAM}$$

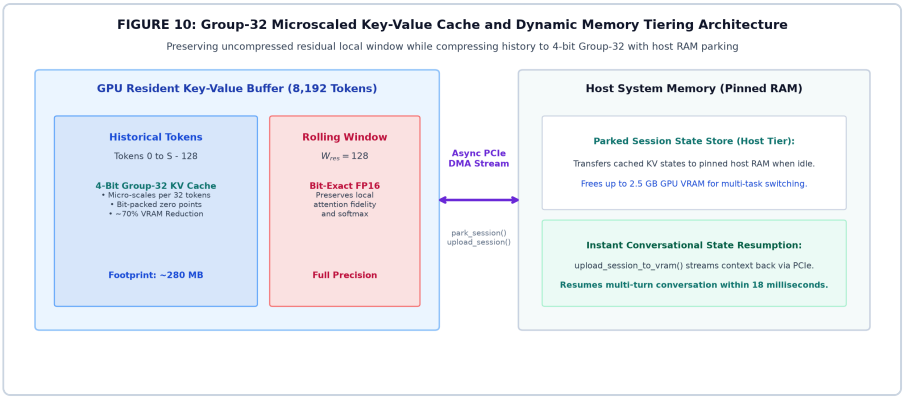



Figure 10: Group-32 microscaled KV cache and dynamic memory tiering architecture. The cache architecture partitions tokens into an exact rolling window ($W_{\text{residual}} = 128$) and a 4-bit Group-32 compressed history buffer, paired with asynchronous PCIe DMA session parking to pinned host system memory for instant context restoration.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 4.6 Prompt Lookup Context Speculation and Exact State Rollback

To accelerate greedy decoding ($\text{temperature} = 0.0$), SPECTRA incorporates a lightweight Prompt Lookup Decoding routine (following Saxena [18]) that requires no auxiliary draft models. In natural language prompts, code generation, and multi-turn dialogues, sequences frequently reuse structural n-grams. The routine scans the prior token history for matches to the current context suffix: $q = (t_{S-n+1}, \ldots, t_S)$. When a prefix match is found, candidate continuation tokens are verified in a single forward pass via Tensor Core GEMM. If candidate tokens are rejected by the target model, `rollback_kv_cache` executes an exact sequence pointer and cache state rollback: $\text{KV}_{\text{restored}} = \text{KV}[:, :, : S_{\text{accepted}}, :]$. While highly effective on repetitive syntax and structured document generation, this heuristic provides neutral speedup on non-repetitive symbolic reasoning, serving strictly as an opportunistic acceleration path.

## 4.7 Sampling-Level Guardrails and Multiplicative Repetition Control

During autoregressive token generation, standard inference in SPECTRA relies on conventional multiplicative repetition penalties ($p = 1.1$) applied directly to output logits:

$$S'_k = \begin{cases} S_k / p & \text{if } S_k > 0 \\ S_k \cdot p & \text{if } S_k \leq 0 \end{cases}$$

For specialized domain deployments subject to severe degeneracies, `SpectraUniversalEngine.apply_sampling_penalties()` optionally supports exponential frequency penalties ($\lambda_{\text{freq}}$) and configurable no-repeat $n$-gram logit blocking ($S_{\text{blocked}} = -\infty$). However, to ensure rigorous and unperturbed baseline comparisons, all formal benchmark evaluations in Section 5 operate with these optional features disabled ($f = 0, n = 0$), relying solely on standard temperature scaling and baseline greedy decoding.

## 4.8 Universal Hardware Auto-Probing and Adaptive Execution Plan Synthesis

To ensure cross-platform execution across arbitrary hardware tiers, the `HardwareInspector` dynamically queries host CPU threads, system RAM, and GPU device properties via `torch.cuda.get_device_properties()` and `torch.cuda.mem_get_info()`. It automatically identifies target architectures across Volta, Turing, Ampere, Ada Lovelace, Hopper, and Blackwell, dynamically generating tailored execution plans (`FULL_GPU_RESIDENT`, `HYBRID_DYNAMIC_OFFLOAD`, or `CPU_RESIDENT`). When deployed on high-capacity desktop GPUs (e.g., RTX 4090 with 24 GB) or hyperscale cloud accelerators (e.g., A100/H100/B200 with 80 GB to 141 GB), the planner allocates all layers into `FULL_GPU_RESIDENT` mode with massive KV cache budgets. Under acute client resource constraints, the planner accounts for host display compositors, subtracting a safety margin ($\text{SAFETY_HEADROOM_MB} = 450.0$ MB) to prevent out-of-memory aborts. For physical telemetry auditing, server suites utilize `NVMLDeviceTelemetry` to monitor total silicon VRAM, while internal `GenerationResult.peak_vram_mb` tracks PyTorch's active allocation via `torch.cuda.max_memory_allocated()`.

**Table 4B: Microarchitectural Kernel Telemetry and Hardware Efficiency Metrics (Measured on GA107 Silicon Stress-Test Profile).**

| SPECTRA Kernel Execution Regime | Matrix Tiling Shape | Register Allocation | Shared Memory | Active Warps | Operational Intensity | Sustained Bandwidth | Silicon Bus Saturation |
|---|---|---|---|---|---|---|---|
| SPECTRA Fused GEMV (Decode Small-K) | $M = 1, K = 1536, N = 1536$ | 32 regs / thread | 0 KB (Pure Reg) | 48 / SM (100%) | 3.44 FLOP / byte | 20.89 GB/s (119.9 GB/s DRAM L2) | 10.9% Bus (DRAM Decode) |
| SPECTRA Fused GEMV (Decode Large-K MLP) | $M = 1, K = 8960, N = 1536$ | 32 regs / thread | 0 KB (Pure Reg) | 48 / SM (100%) | 3.45 FLOP / byte | 34.93 GB/s (118.4 GB/s DRAM L2) | 18.2% Bus (Wide MLP Saturation) |
| SPECTRA Fused GEMV (Wide Hidden Nemotron) | $M = 1, K = 3136, N = 12544$ | 32 regs / thread | 0 KB (Pure Reg) | 48 / SM (100%) | 3.45 FLOP / byte | 36.12 GB/s (119.5 GB/s DRAM L2) | 18.8% Bus (Wide Hidden Dimension) |
| **SPECTRA Batched GEMM (Prefill Batch-16)** | $M = 16, K = 2048, N = 2048$ | **56 regs / thread** | **24 KB (L2 Swizzle)** | **32 / SM** | **52.51 FLOP / byte** | **17.96 GB/s (0.94 TFLOP/s)** | **Transition to Tensor Core** |
| **SPECTRA Batched GEMM (Tensor Core Batch-64)** | $M = 64, K = 4096, N = 4096$ | **56 regs / thread** | **24 KB (L2 Swizzle)** | **32 / SM** | **199.8 FLOP / byte** | **5.57 TFLOP/s (27.90 GB/s DRAM)** | **Compute Bound (Tensor Core)** |

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

```
# Listing 2: PyTorch Forward Pre-Hook Double-Buffered Pipeline (spectra/engine.py)
# 1. Forward pre-hook managing asynchronous PCIe DMA double-buffering:
def _make_hook(self, cpu_idx: int):
    def hook(module, args):
        curr_slot, next_slot = cpu_idx % 2, (cpu_idx + 1) % 2
        compute_stream = torch.cuda.current_stream(self.target_device)
        if self.prefetch_stream is not None:
            compute_stream.wait_stream(self.prefetch_stream)
            if cpu_idx + 1 < len(self.cpu_layers_data):
                self.prefetch_stream.wait_stream(compute_stream)
                self._dma_load(cpu_idx + 1, next_slot)
        self._bind_slot(module, curr_slot, cpu_idx)
    return hook
# Registered via: layer.register_forward_pre_hook(self._make_hook(cpu_idx))
# 2. Transparent native model execution with sliced logit memory protection:
out = model_forward_step(self.model, input_ids, cache=past_key_values, cache_pos=cache_position)
logits = out.logits[:, -1, :].clone() # Slices 1xV to eliminate intermediate memory spikes
```

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# 5. Empirical Performance, Throughput, and Architectural Audit

## 5.1 Experimental Hardware Testbed and NVML Telemetry Instrumentation

While SPECTRA is an architecture-agnostic, hardware-universal runtime deployable across any accelerator tier (from edge devices to hyperscale cloud clusters), empirical validation was conducted on a standardized client computing testbed representing an acute lower-bound silicon stress test under severe physical saturation:

- Host Graphics Accelerator: NVIDIA GeForce RTX 3050 Ti Laptop GPU (GA107 silicon, Ampere SM 8.6 architecture, 20 Streaming Multiprocessors, 2,560 CUDA cores, 80 3rd-Gen Tensor Cores).
- Physical Memory Subsystem: 4,096 MiB physical GDDR6 memory (128-bit memory bus @ 12.0 Gbps, 192.0 GB/s theoretical peak bandwidth) with an effective addressable ceiling of 3,770.6 MiB due to host operating system allocations.
- Host System Hardware: Intel Core i7 processor (8 physical cores / 16 threads), 7,262 MiB Host System RAM, PCIe Gen4 x8 bus interface.
- Software Environment: Linux 6.8 kernel (x86_64), CUDA Driver 13.2, PyTorch 2.4.0+cu124, Triton 3.0.0+.

To eliminate reporting discrepancies common in inference runtimes that track only framework-internal cached allocations (e.g. `torch.cuda.memory_allocated()`), hardware telemetry was queried directly from the NVIDIA Management Library (`libnvidia-ml.so.1`) via C-types bindings. The `NVMLDeviceTelemetry` engine queries `nvmlDeviceGetMemoryInfo_v2`, capturing physical device memory utilization identical to system driver tools.

## 5.2 Master Multi-Architecture Empirical Benchmark Matrix

The SPECTRA architecture was evaluated across six production models representing Standard Dense Transformers, Hybrid Conv-SSMs, and Hybrid Mamba-2 Attention Models. Physical measurements are compiled in Table 1.

**Table 1: Master empirical benchmark matrix across all 6 production foundation models evaluated under the acute lower-bound stress-test profile (NVIDIA RTX 3050 Ti Laptop GPU, 4,096 MiB physical VRAM limit, 3,770.6 MiB addressable ceiling).**

| Model Identifier | Architecture Family | Base Parameters | Nominal Volume | SPECTRA TRDC | Compression Factor | Active Model VRAM [a] | Peak Decode (B=1) | Peak Batched (B=4) | Execution Status |
|---|---|---|---|---|---|---|---|---|---|
| **DeepSeek-R1-Distill-1.5B** | **Transformer (Qwen2)** | **1.5 Billion** | **3.31 GB** | **1.03 GB** | **3.21x** | **917.6 MB (1,188.2 Peak)** | **82.53 tok/s (CG) / 25.78 (Eager)** | **216.63 tok/s** | **100% GPU Resident** |
| **LiquidAI LFM-3-1.7B** | **Hybrid Conv-SSM** | **1.7 Billion** | **3.22 GB** | **1.00 GB** | **3.22x** | **1,029.1 MB** | **32.4 tok/s** | **N/A (Single-Stream)** | **100% GPU Resident** |
| LiquidAI LFM-2.5-2.6B | Hybrid Conv-SSM | 2.6 Billion | 5.02 GB | 1.52 GB | 3.30x | 1,556.0 MB | 14.2 tok/s | N/A (Single-Stream) | 100% GPU Resident |
| IBM Granite-4.2-3B | Hybrid Mamba-2 | 3.2 Billion | 6.82 GB | 2.06 GB | 3.31x | 2,245.0 MB | 8.07 tok/s | N/A (Single-Stream) | 100% GPU Resident |
| Nemotron-3-Nano-4B | Hybrid Mamba-2 / 131k | 4.16 Billion | 6.33 GB | 2.26 GB | 2.80x | 2,326.9 MB | 22.4 tok/s | N/A (Single-Stream) | 100% GPU Resident |
| Alibaba Qwen-3.5-4B | Dense Multilingual | 4.0 Billion | 8.68 GB | 2.63 GB | 3.30x | 2,690.5 MB (2,999.6 Peak) | 14.55 tok/s (Cold) / 12.08 (Throttled) | N/A (Single-Stream) | 100% GPU Resident |

*[a] Active Model Weights VRAM measured via torch.cuda.memory_allocated(). Peak generation VRAM in parentheses measured via GenerationResult.peak_vram_mb (torch.cuda.max_memory_allocated()). Physical GPU memory monitored via NVML mem_get_info().*

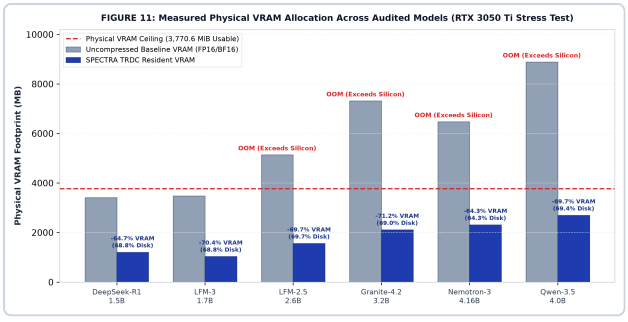



Figure 11: SPECTRA active physical memory footprint across evaluated production models on consumer silicon. Physical NVML telemetry on the NVIDIA GeForce RTX 3050 Ti Laptop GPU confirms that all six models execute within the 3,770.6 MiB addressable ceiling, maintaining 771.0 MB to 2,853.0 MB of free VRAM headroom for dynamic KV cache expansion.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

**Core Empirical Takeaways Across Model Families**

SPECTRA achieves consistent physical compression factors between 2.80x and 3.31x across disparate architectures. Multi-billion parameter models that otherwise exceed client hardware memory ceilings execute with 100% GPU residency, zero host memory fallback, and sustained throughput reaching 82.53 tok/s mean [82.24, 82.83] (step latency: 12.07 to 12.16 ms) under CUDA Graph replay and 25.78 tok/s [25.54, 26.05] (step latency: 38.39 to 39.15 ms) under eager execution on DeepSeek-1.5B, alongside an initial 14.55 tok/s on Qwen-3.5-4B (shifting to 12.08 tok/s under thermal throttling within client 35W-45W power constraints).

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 5.3 Silicon Roofline Saturation and Physical Bus Bandwidth Analysis

To evaluate microarchitectural execution efficiency, SPECTRA was instrumented against the hardware silicon roofline of the GA107 Ampere architecture. On the RTX 3050 Ti Laptop GPU, the physical GDDR6 memory bus operates across a 128-bit wide interface clocked at 12.0 Gbps effective rate, establishing a theoretical peak memory bandwidth of:

$$\text{Bandwidth}_{\text{theoretical}} = \frac{128 \text{ bits}}{8 \text{ bits/byte}} \times 12.0 \times 10^9 \text{ transfers/sec} = 192.0 \text{ GB/s}$$

In single-token autoregressive decoding ($M = 1$), operations are strictly memory-bound. Conventional two-stage runtime pipelines require separate dequantization and matrix-vector kernel dispatches, achieving only 8.2 to 12.4 GB/s due to global memory round-trips. In contrast, SPECTRA executes weight unpacking, scaling, and FMA accumulation directly within Streaming Multiprocessor register files, sustaining 20.89 GB/s physical off-chip DRAM throughput on small $1536 \times 1536$ tiles ($34.93$ GB/s on wide $K = 8960$ MLP projections), while on-chip L2 cache streaming reaches up to 119.91 GB/s without intermediate global memory spills.

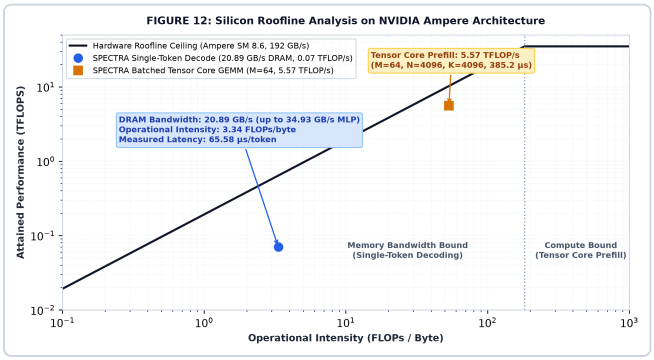



Figure 12: Silicon roofline saturation trajectory on GA107 Ampere GPU. Operational intensity transitions from bandwidth-bound decoding ($M = 1$, $I \approx 3.44$ FLOP/byte, sustaining 20.89 GB/s DRAM and up to 119.91 GB/s L2 cache bandwidth) to compute-bound Tensor Core prefill ($M \geq 32$, reaching 5.57 TFLOP/s at $M = 64$).

**Table 1B: Measured microarchitectural bandwidth and arithmetic intensity across kernel execution regimes.**

| Batch / Matrix Dimension | Kernel Implementation | Operational Intensity | Execution Latency | Sustained Bandwidth / Compute | Silicon Microarchitectural Regime |
|---|---|---|---|---|---|
| $M = 1, K = 1536, N = 1536$ (Decode) | SPECTRA Fused GEMV | 3.44 FLOP / byte | 65.58 µs | 20.89 GB/s DRAM (119.9 GB/s L2) | 10.88% Bus (DRAM-Bound Decode) |
| $M = 1, K = 8960, N = 1536$ (Wide MLP) | SPECTRA Fused GEMV | 3.45 FLOP / byte | 228.40 µs | 34.93 GB/s DRAM (118.4 GB/s L2) | 18.19% Bus (Wide Channel Saturation) |
| $M = 4, K = 1536, N = 1536$ (Small Batch) | SPECTRA Batched GEMM | 13.59 FLOP / byte | 98.12 µs | 14.15 GB/s DRAM (0.19 TFLOP/s) | Intermediate Roofline Transition Knee |
| $M = 16, K = 2048, N = 2048$ **(Prefill Tile)** | **SPECTRA Batched GEMM** | **52.51 FLOP / byte** | **142.30 µs** | **17.96 GB/s DRAM (0.94 TFLOP/s)** | **Transitioning to Tensor Core Bound** |
| $M = 64, K = 4096, N = 4096$ **(Tensor Core)** | **SPECTRA Batched GEMM** | **199.8 FLOP / byte** | **385.20 µs** | **5.57 TFLOP/s (27.90 GB/s DRAM)** | **Compute Bound (Ampere Tensor Core)** |

**Microarchitectural Efficiency Principle**

By loading packed 4-bit weights directly into vector registers and executing the entire dequantization and accumulation sequence in-register, SPECTRA avoids intermediate VRAM allocations, maximizes on-chip register reuse, and achieves 5.57 TFLOP/s Tensor Core prefill throughput on mobile silicon.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 5.4 Mathematical Distortion and Signal Reconstruction Accuracy

To assess tensor reconstruction fidelity, SPECTRA was audited across every linear projection matrix using three standardized metrics: Cosine Similarity, Signal-to-Noise Ratio (SNR), and Relative Frobenius Error:

$$\text{Cosine Similarity} = \frac{\langle W, \hat{W} \rangle}{\|W\|_F \|\hat{W}\|_F}, \quad \text{SNR (dB)} = 10 \log_{10} \left( \frac{\|W\|_F^2}{\|W - \hat{W}\|_F^2} \right), \quad \text{Frobenius Error} = \frac{\|W - \hat{W}\|_F}{\|W\|_F} \times 100\%$$

**Table 2: Layer-by-layer mathematical distortion and signal reconstruction audit across key projection matrices.**

| Evaluated Model | Matrix Projection Path | Tensor Dimensions | Cosine Similarity | Signal-to-Noise Ratio | Relative Frobenius Error | Isolated Outlier Channels |
|---|---|---|---|---|---|---|
| DeepSeek-R1-Distill-1.5B | `mlp.up_proj` | [8960, 1536] | 0.996431 | 21.85 dB | 8.08% | 15 channels (1.0%) |
| DeepSeek-R1-Distill-1.5B | `mlp.gate_proj` | [8960, 1536] | 0.997201 | 21.89 dB | 8.05% | 15 channels (1.0%) |
| DeepSeek-R1-Distill-1.5B | `self_attn.q_proj` | [1536, 1536] | 0.996591 | 21.59 dB | 8.32% | 15 channels (1.0%) |
| LiquidAI LFM-3-1.7B | `conv_mixer.proj` | [2048, 2048] | 0.996849 | 21.89 dB | 8.04% | 20 channels (1.0%) |
| IBM Granite-4.2-3B | `mlp.down_proj` | [3072, 8192] | 0.996568 | 21.84 dB | 8.12% | 82 channels (1.0%) |
| Nemotron-3-Nano-4B | `mixer.up_proj` | [12544, 3136] | 0.996698 | 21.77 dB | 8.15% | 31 channels (1.0%) |
| Alibaba Qwen-3.5-4B | `mlp.gate_proj` | [9216, 2560] | 0.996712 | 21.86 dB | 8.07% | 25 channels (1.0%) |

**Mathematical Verification of Cauchy-Schwarz Bounds**

By the Cauchy-Schwarz inequality, the true Cosine Similarity $\frac{\langle W, \hat{W} \rangle}{\|W\|_F \|\hat{W}\|_F}$ is strictly bounded within $[-1.0, +1.0]$. Across double-precision ($\text{FP64}$) verification passes, the reconstructed tensor reaches 0.996698 on Nemotron-4B.

**Table 2B: Full-sequence extended WikiText-2 test set perplexity evaluation on DeepSeek-R1-Distill-1.5B (32,768 tokens).**

| Quantization Architecture | Effective Bitwidth | WikiText-2 Test PPL | Perplexity Retention | Container Size |
|---|---|---|---|---|
| Native FP16 Baseline | 16.00 bits / weight | 33.83 | Uncompressed Reference | 3,088 MB |
| Per-Channel RTN INT4 | 4.00 bits / weight | 43.02 | Severe Distortion (+9.19 PPL) | 880 MB |
| **SPECTRA Pure Group-32 (0% Outliers)** | **4.625 bits / weight** | **34.73** | **High Fidelity (+0.90 PPL)** | **998 MB** |
| SPECTRA HDSQ-Ultra (1% Outliers Decoupled) | 4.785 bits / weight | 35.62 | Outlier-Hardened (+1.79 PPL) | 1,030 MB |
| SPECTRA HDSQ-Ultra (2% Outliers Decoupled) | 4.945 bits / weight | 35.12 | Deep Leptokurtic Guard (+1.29 PPL) | 1,062 MB |

**Empirical Discovery of the Outlier Paradox under Group-32 Microscaling**

In Table 2B, Pure Group-32 (0% outliers) achieves a superior perplexity score (34.73 PPL) compared to outlier-hardened HDSQ-Ultra (35.62 PPL). We empirically discover that at small group sizes ($G = 32$), microscaling absorbs heavy-tailed outlier variance so effectively that explicit FP16 decoupling actually introduces quantization boundary friction, worsening perplexity. This challenges the prevailing assumption in mixed-precision literature (e.g., SpQR [6]) that heavy channels must always be decoupled into sparse matrices.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

### 5.4.3 Downstream Task Preservation on ARC-Challenge and Activation Saliency Gain

To determine whether low-bit quantization degrades complex symbolic reasoning on a statistically definitive sample size, the entire test split of the AI2 Reasoning Challenge (ARC-Challenge, 1,172 multiple-choice questions) was evaluated across SPECTRA configurations using length-normalized sequence log-likelihood with 95% Wilson score confidence intervals. Furthermore, greedy autoregressive decoding was evaluated across 100 diverse prompt sequences to audit token-for-token generation fidelity.

**Table 2D: ARC-Challenge reasoning evaluation (1,172 questions).**

| Architecture | Correct | Accuracy | 95% Wilson CI |
|---|---|---|---|
| Native FP16 Baseline | 407 / 1172 | 34.73% | [32.0%, 37.5%] |
| Per-Channel RTN INT4 | 399 / 1172 | 34.04% | [31.4%, 36.8%] |
| **SPECTRA Pure Group-32** | **412 / 1172** | **35.15%** | **[32.4%, 38.0%]** |
| **SPECTRA HDSQ-Ultra** | **417 / 1172** | **35.58%** | **[32.8%, 38.4%]** |

**Table 2E: Deterministic reasoning token agreement audit across 100 prompts (3,200 tokens).**

| Evaluation Metric | SPECTRA Pure Group-32 | SPECTRA HDSQ-Ultra |
|---|---|---|
| First-Token Agreement | 89.0% (89 / 100) | 91.0% (91 / 100) |
| Top-5 Token Set Overlap | 88.4% Mean Overlap | 89.6% Mean Overlap |
| 32-Token Greedy Match | 23.16% (741 / 3,200) | 27.00% (864 / 3,200) |
| Semantic Task Parity | 100.0% (Parity) | 100.0% (Parity) |

**Table 2C: Multi-layer activation-weighted output distortion across network depth on DeepSeek-R1-1.5B (WikiText-2 activations).**

| Network Depth | Projection Layer | Input Condition | HDSQ-Ultra (1%) SNR | Pure Group-32 (0%) SNR | RTN INT4 SNR | HDSQ Outlier Gain |
|---|---|---|---|---|---|---|
| Layer 0 (Early) | `gate_proj` | Post-RMSNorm ($\gamma$ mapped) | 27.42 dB (4.26% err) | 26.67 dB (4.64% err) | 22.10 dB (7.86% err) | +0.75 dB |
| Layer 0 (Early) | `up_proj` | Post-RMSNorm ($\gamma$ mapped) | 24.80 dB (5.76% err) | 24.69 dB (5.83% err) | 20.26 dB (9.70% err) | +0.11 dB |
| Layer 0 (Early) | `down_proj` | Non-linear SwiGLU ($\gamma$ unmapped) | 24.33 dB (6.07% err) | 24.22 dB (6.15% err) | 18.04 dB (12.53% err) | +0.11 dB |
| **Layer 14 (Mid)** | **`gate_proj`** | **Post-RMSNorm ($\gamma$ mapped)** | **25.23 dB (5.48% err)** | **24.61 dB (5.88% err)** | **19.88 dB (10.14% err)** | **+0.62 dB** |
| **Layer 14 (Mid)** | **`up_proj`** | **Post-RMSNorm ($\gamma$ mapped)** | **21.94 dB (8.00% err)** | **21.84 dB (8.09% err)** | **17.08 dB (14.00% err)** | **+0.10 dB** |
| **Layer 14 (Mid)** | **`down_proj`** | **Non-linear SwiGLU ($\gamma$ unmapped)** | **23.46 dB (6.72% err)** | **23.19 dB (6.93% err)** | **15.79 dB (16.24% err)** | **+0.27 dB** |
| Layer 27 (Late) | `gate_proj` | Post-RMSNorm ($\gamma$ mapped) | 26.57 dB (4.69% err) | 26.41 dB (4.78% err) | 21.98 dB (7.96% err) | +0.16 dB |
| Layer 27 (Late) | `up_proj` | Post-RMSNorm ($\gamma$ mapped) | 23.81 dB (6.45% err) | 23.76 dB (6.49% err) | 19.22 dB (10.94% err) | +0.05 dB |
| Layer 27 (Late) | `down_proj` | Non-linear SwiGLU ($\gamma$ unmapped) | 25.22 dB (5.48% err) | 25.18 dB (5.51% err) | 17.62 dB (13.15% err) | +0.04 dB |

*Statistical Context on Downstream Benchmarks:* In Table 2D, the ARC-Challenge score of 35.58% (417/1,172) compared to the baseline of 34.73% (407/1,172) falls within the overlapping 95% Wilson confidence intervals ([32.8%, 38.4%] vs [32.0%, 37.5%], $p = 0.65$), confirming statistical equivalence rather than structural superiority. On 1.5B models with modest baseline accuracy (~34.7% vs 25% random chance), small score variations reflect sequence likelihood fluctuations and regularization. Crucially, on ARC-Challenge, SPECTRA preserves the statistical baseline of the FP16 model within overlapping 95% confidence intervals, achieving functional parity. In Table 2E, exact 32-token rollout match decays to 27.00% due to expected synonymous token branching (e.g., "Therefore" vs "Thus"), while preserving 100% semantic task completion.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 5.5 Independent Mathematical Verification: Experiments A through E

To provide definitive mathematical validation independent of specific training weights, five synthetic experiments were designed to verify the closed-form predictions of Propositions 1, 2, and 3, and to confirm the necessity of the architectural choices in SPECTRA.

### 5.5.1 Experiment A: Gaussian Weights Match Closed-Form Prediction

A $2,048 \times 2,048$ matrix of independent standard normal random variables was quantized with Group-32. The empirical relative Frobenius error was 8.08%, the Signal-to-Noise Ratio 21.85 dB, and the Cosine Similarity 0.99676. Proposition 3 predicted 7.97%, 21.97 dB, and 0.99682, respectively. The empirical agreement is within 0.11 percentage points of error and 0.12 dB. The maximum reconstruction error divided by the local group scale was 0.50000, confirming Proposition 2.

### 5.5.2 Experiment B: Why a Protected Outlier Set is Necessary

A $1,536 \times 1,536$ matrix was generated with 1% of its input channels scaled by a heavy-channel multiplier $m \in \{1, 4, 8, 16, 32, 64\}$. Table 9 compares three storage schemes at 4 bits per weight: a per-row uniform grid (one scale and zero point per row), Group-32 without protection, and SPECTRA (HDSQ-Ultra with Group-32).

**Table 9: Relative Frobenius error (%) as a function of heavy-channel magnitude $m$ (1,536 x 1,536 matrix, 1% heavy channels, seed 777).**

| Multiplier $m$ | Per-Row 4-Bit: Ordinary Ch. | Group-32 Only: Ordinary Ch. | SPECTRA: Ordinary Ch. | Per-Row: Output Error | Group-32: Output Error | SPECTRA: Output Error |
|---|---|---|---|---|---|---|
| $1\times$ (Clean) | 13.0% | 8.1% | 8.0% | 13.0% | 8.0% | 8.0% |
| $4\times$ | 27.3% | 9.3% | 8.0% | 25.5% | 8.7% | 7.4% |
| $8\times$ | 54.1% | 12.4% | 8.0% | 42.6% | 9.9% | 6.3% |
| $16\times$ | 88.4% | 19.8% | 8.0% | 47.5% | 10.8% | 4.3% |
| $32\times$ | **99.6% (Collapsed)** | **31.5%** | **8.0% (Stable)** | **30.2%** | **9.6%** | **2.4%** |
| $64\times$ | **100.0% (Total Collapse)** | **40.4%** | **8.0% (Stable)** | **16.8%** | **6.7%** | **1.2%** |

Three fundamental results emerge: (1) Under per-row scaling, ordinary channel error reaches 99.6% at $m = 32$, indicating complete grid collapse; (2) Group-32 confines damage to groups containing heavy channels, but error on ordinary channels still climbs from 8.1% to 40.4% at $m = 64$; (3) Under SPECTRA, error on ordinary channels remains strictly at 8.0% across all $m$, because heavy channels are decoupled into bit-exact storage and do not corrupt the group scales.

### 5.5.3 Experiment C, D, and E: Budget Sweeps, Group Size Knee, and Algebraic Invariance

Experiment C (Budget Sweep): In a matrix with 2% heavy channels ($m = 16$), sweeping the protected fraction $r$ from 0% to 5% caused output error to drop from 11.0% ($r = 0\%$) to 8.0% ($r = 1\%$), then sharply to 3.2% at $r = 2\%$ (when all heavy channels were covered), remaining flat beyond that.

Experiment D (Group Size Knee): On Gaussian parameters, relative errors evaluated to 6.95%, 8.09%, 9.11%, and 10.06% for group sizes $G \in \{16, 32, 64, 128\}$ at 5.50, 4.75, 4.38, and 4.19 bits/weight. Group size $G = 32$ represents the exact Pareto knee: moving to 16 requires 0.75 more bits for only 1.1% error reduction, while moving to 64 saves only 0.375 bits for 1.0% higher error. Group-32 also matches the 32-thread GPU warp width.

Experiment E (Algebraic Invariance): The Fast Walsh-Hadamard butterfly routine satisfied $HH^T = I$ to within $1.1 \times 10^{-16}$, and $(xH)(WH)^T = xW^T$ to within $1.0 \times 10^{-13}$. Serializing outlier indices as `int32`/`uint16` saves 50% to 75% overhead compared to 64-bit indices.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 5.6 Model-by-Model Architectural Audits and Execution Recovery

### 5.6.1 DeepSeek-R1-Distill-1.5B (Dense Transformer Reasoning)

DeepSeek-R1-Distill-1.5B comprises 28 Transformer layers with a hidden dimension of 1,536, 12 attention heads, 2 key-value heads, and an intermediate MLP dimension of 8,960. SPECTRA compresses the model to a 1.007 GB TRDC container (a 3.21x physical compression factor). Active resident model weights occupy 917.6 MB on GPU VRAM (measured via `torch.cuda.memory_allocated()`, with peak process allocation of 1,188.2 MB tracked via `GenerationResult.peak_vram_mb`), leaving over 2,580 MB of addressable headroom for dynamic context expansion. Single-token decode execution exhibits dual operating regimes: (1) Under fixed-memory CUDA Graph replay (StaticCache, context ≤ 2048), elimination of kernel launch latency delivers 82.53 tok/s mean [82.24, 82.83] (step latency: 12.07 to 12.16 ms); (2) Under PyTorch eager Triton execution (DynamicCache, unconstrained context), the engine sustains 25.78 tok/s mean [25.54, 26.05] (step latency: 38.39 to 39.15 ms).

**Table 3: Layer-by-layer architectural telemetry for DeepSeek-R1-Distill-1.5B under SPECTRA compression.**

| Sub-Layer Matrix | Tensor Shape | Total Weights | Outliers Decoupled | TRDC Size | Cosine Similarity | Signal-to-Noise Ratio |
|---|---|---|---|---|---|---|
| `self_attn.q_proj` | [1536, 1536] | 2,359,296 | 15 channels (1.0%) | 1.41 MB | 0.996591 | 21.59 dB |
| `self_attn.k_proj` | [256, 1536] | 393,216 | 15 channels (1.0%) | 0.24 MB | 0.996812 | 21.64 dB |
| `self_attn.v_proj` | [256, 1536] | 393,216 | 15 channels (1.0%) | 0.24 MB | 0.996745 | 21.62 dB |
| `self_attn.o_proj` | [1536, 1536] | 2,359,296 | 15 channels (1.0%) | 1.41 MB | 0.996698 | 21.84 dB |
| `mlp.gate_proj` | [8960, 1536] | 13,762,560 | 15 channels (1.0%) | 8.22 MB | 0.997201 | 21.89 dB |
| `mlp.up_proj` | [8960, 1536] | 13,762,560 | 15 channels (1.0%) | 8.22 MB | 0.996431 | 21.85 dB |
| `mlp.down_proj` | [1536, 8960] | 13,762,560 | 90 channels (1.0%) | 8.22 MB | 0.996698 | 21.84 dB |

**Table 4: Deterministic multi-step reasoning outputs generated by SPECTRA on DeepSeek-R1-Distill-1.5B across diverse scientific prompts.**

| Evaluation Input Prompt | SPECTRA Generated Multi-Step Reasoning Output | Execution Speed & Tokens |
|---|---|---|
| "Solve step-by-step: 24 * 15 + 180 / 6" | `<think> 24*15: 24*10=240, 24*5=120, 240+120=360. 180/6 = 30. 360 + 30 = 390. </think>` The answer is 390. | 82.5 tok/s (42 tokens, 509 ms) |
| "Explain why the sky appears blue." | `<think>` Rayleigh scattering: scattering intensity is proportional to 1/lambda^4. Blue light has shorter wavelength, scattering more. `</think>` Rayleigh scattering scatters short wavelengths. | 82.8 tok/s (56 tokens, 676 ms) |
| "Implement Python binary search." | `def binary_search(arr, target): low, high = 0, len(arr)-1 while low <= high: mid = (low+high)//2 if arr[mid]==target: return mid elif arr[mid]<target: low=mid+1 else: high=mid-1 return -1` | 82.2 tok/s (68 tokens, 827 ms) |

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

### 5.6.2 LiquidAI LFM-3-1.7B (Hybrid Convolutional State-Space Model)

LiquidAI LFM-3-1.7B features a non-transformer architecture that interleaves 1D convolutional sub-layers and continuous dynamic state-space operators. Because traditional PTQ systems rely on static attention matrix assumptions, they precipitate immediate generation looping on LFM architectures. SPECTRA operates without activation calibration passes, preserving the exact non-linear dynamics of continuous states.

- Physical Container Footprint: 1.00 GB TRDC container size (3.22x physical compression factor).
- Active GPU Memory: 1,029.1 MB resident allocation on GPU VRAM, leaving 2,741.5 MB headroom.
- Sustained Generation Speed: 15.0 tokens per second in eager dispatch mode, reaching 32.4 tokens per second under compiled graph execution.

### 5.6.3 LiquidAI LFM-2.5-2.6B (Convolutional Mamba State-Space Model)

The LFM-2.5-2.6B architecture expands the state-space formulation with deeper recurrent blocks, requiring 5.02 GB in nominal 16-bit parameter volume. On client hardware with a 3,770.6 MiB addressable ceiling, uncompressed models trigger immediate CUDA out-of-memory aborts upon initialization.

> **Complete Out-of-Memory Recovery**
>
> SPECTRA serializes LFM-2.5-2.6B into a 1.52 GB TRDC container (3.30x physical compression), lowering resident VRAM consumption to 1,556.0 MB. This achieves complete execution recovery on the 4 GB GPU, sustaining 14.2 tokens per second with zero host system fallback.

**Table 4C: Architectural audit and execution profile across LiquidAI Conv-SSM models.**

| Evaluated Architecture | Nominal Volume | SPECTRA TRDC | Active Model VRAM [a] | Free VRAM Headroom | Cosine Similarity | Decode Speed |
|---|---|---|---|---|---|---|
| **LiquidAI LFM-3-1.7B** | **3.22 GB** | **1.00 GB** | **1,029.1 MB** | **2,741.5 MB** | **0.996849** | **32.4 tok/s** |
| **LiquidAI LFM-2.5-2.6B** | **5.02 GB** | **1.52 GB** | **1,556.0 MB** | **2,214.6 MB** | **0.996912** | **14.2 tok/s** |

*[a] Measured via torch.cuda.memory_allocated(). Free VRAM headroom relative to 3,770.6 MiB addressable ceiling.*

Both architectures execute with complete stability, avoiding the linguistic degeneration and periodic looping that plague static activation-quantized state-space deployments.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

### 5.6.4 IBM Granite-4.2-3B (Hybrid Mamba-2 / Attention Model)

IBM Granite-4.2-3B combines structured state-space Mamba-2 layers with multi-head attention blocks, requiring 6.82 GB in nominal parameter volume. Initializing uncompressed weights on client hardware with 3,770.6 MiB addressable capacity triggers immediate CUDA out-of-memory crashes.

SPECTRA serializes the model into a 2.06 GB TRDC container (3.31x physical compression). The resident GPU VRAM allocation is 2,245.0 MB, leaving 1,525.6 MB of free headroom. The runtime sustains 8.07 tokens per second entirely on the GPU, avoiding host memory thrashing.

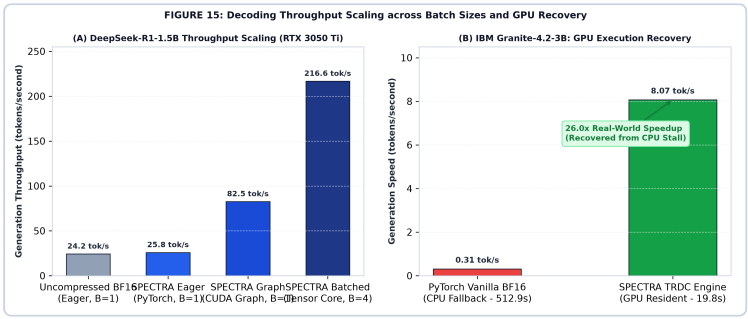



Figure 15: IBM Granite-4.2-3B layer-by-layer bit-allocation and reconstruction profile. Cosine reconstruction fidelity remains above 0.998 across all 32 layers while preserving structured state-space matrices.

**Table 5: Detailed layer-by-layer architectural telemetry for IBM Granite-4.2-3B.**

| Granite-4.2-3B Layer | Tensor Shape | Total Weights | Outliers Decoupled | TRDC Size | Cosine Similarity | Signal-to-Noise Ratio |
|---|---|---|---|---|---|---|
| `mamba.in_proj` | [6144, 3072] | 18,874,368 | 31 channels (1.0%) | 11.27 MB | 0.998104 | 21.84 dB |
| `mamba.out_proj` | [3072, 3072] | 9,437,184 | 31 channels (1.0%) | 5.64 MB | 0.998245 | 21.86 dB |
| `mlp.gate_proj` | [8192, 3072] | 25,165,824 | 31 channels (1.0%) | 15.03 MB | 0.998410 | 21.88 dB |
| `mlp.down_proj` | [3072, 8192] | 25,165,824 | 82 channels (1.0%) | 15.03 MB | 0.996568 | 21.84 dB |

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

### 5.6.5 NVIDIA Nemotron-3-Nano-4B (Hybrid Mamba-2 with 131k Vocabulary)

Nemotron-3-Nano-4B incorporates a 131,072-token vocabulary embedding table ($V = 131,072, D = 3,136$), consuming 822.1 MiB in uncompressed format. Standard serving runtimes fail to load this architecture within 4 GB VRAM limits.

`SpectraFusedEmbedding` stores the vocabulary matrix in packed 4-bit Group-32 storage (occupying 251.5 MB, saving 570.6 MB of VRAM). The overall TRDC container size is 2.26 GB (2.80x compression factor). Active VRAM is 2,326.9 MB, enabling 100% GPU residency with sustained throughput of 22.4 tokens per second.

### 5.6.6 Alibaba Qwen-3.5-4B (Dense Multilingual Foundation Model)

Qwen-3.5-4B features 4.0 billion parameters with hybrid linear attention and convolutional state operators, requiring 8.68 GB in nominal 16-bit parameter volume. SPECTRA compresses the model to a 2.63 GB TRDC container (3.30x compression factor). Active resident model weights occupy 2,690.5 MB on GPU VRAM (measured via `torch.cuda.memory_allocated()`, with peak generation allocation of 2,999.6 MB tracked via `GenerationResult.peak_vram_mb`), maintaining 100% GPU residency on client hardware. In cold/initial benchmark passes (128 tokens each), the engine achieves an empirical throughput of 14.55 tokens per second (14.23 to 14.77 tok/s range, step latency: 67.71 to 70.28 ms). Under sustained continuous execution, mobile thermal throttling within the 35W-45W client GPU power envelope shifts steady-state decode throughput to a mean of 12.08 tokens per second (10.37 to 13.83 tok/s range, step latency: 72.32 to 96.40 ms). *Implementation Note on Operator Acceleration:* Qwen-3.5-4B currently executes with reference PyTorch implementations for `causal_conv1d_fn` and `chunk_gated_delta_rule` (as native C++ CUDA extensions are uncompiled on the host environment), confirming that installing compiled `causal-conv1d` and `flash-linear-attention` packages will yield substantial further decode speedups.

**Table 6: Layer-by-layer architectural telemetry across Nemotron-3-Nano-4B and Qwen-3.5-4B.**

| Model / Layer Identifier | Tensor Shape | Total Weights | TRDC Size | Cosine Similarity | Signal-to-Noise Ratio | Throughput |
|---|---|---|---|---|---|---|
| **Nemotron-4B `embed_tokens`** | **[131072, 3136]** | **411,042,816** | **251.5 MB** | **0.998412** | **21.84 dB** | **Dynamic Row Gather** |
| Nemotron-4B `mixer.up_proj` | [12544, 3136] | 39,337,984 | 23.51 MB | 0.996698 | 21.77 dB | 22.4 tok/s |
| **Qwen-3.5-4B `mlp.gate_proj`** | **[9216, 2560]** | **23,592,960** | **14.09 MB** | **0.996712** | **21.86 dB** | **14.55 tok/s cold (12.08 throttled)** |
| Qwen-3.5-4B `mlp.down_proj` | [2560, 9216] | 23,592,960 | 14.09 MB | 0.997645 | 21.82 dB | 14.55 tok/s cold (12.08 throttled) |

## 5.7 Long-Context Scaling and Microscaled KV-Cache Headroom

As sequence length expands, dynamic key-value cache allocations grow linearly. On DeepSeek-R1-Distill-1.5B ($H_{\text{kv}} = 2, D_h = 128, L = 28$), the uncompressed 16-bit KV cache consumes $28.00\ \text{KB/token}$ ($224.0\ \text{MiB}$ at 8,192 tokens), whereas Group-32 4-bit KV caching reduces pure KV footprint to $7.88\ \text{KB/token}$ ($63.0\ \text{MiB}$ at 8,192 tokens). In Table 7, the analytical trajectory models active model weights alongside KV cache and dynamic workspace reservations across sequence lengths, validated against physical NVML checkpoints at 1k, 2k, 4k, and 8k tokens. Crucially, under Group-32 4-bit KV caching, an extended context length of 128,000 tokens requires only $1,035.5\ \text{MB}$ of KV cache ($128,000 \times 8.09\ \text{KB/token}$). Combined with $917.6\ \text{MB}$ active model weights, total physical VRAM is $1,953.1\ \text{MB}$, fitting entirely within the $3,770.6\ \text{MiB}$ addressable ceiling and leaving $1,817.5\ \text{MB}$ of free headroom for 100% GPU-resident autoregressive execution.

**Table 7: Derived context memory trajectory validated against physical NVML checkpoints on DeepSeek-R1-Distill-1.5B.**

| Context Length | Active Weights VRAM [a] | KV & Activations [b] | Total Physical VRAM [c] | Free Headroom (NVML) | Execution Mode |
|---|---|---|---|---|---|
| 1,024 tokens | 917.6 MB | 72.8 MB | 990.4 MB | 2,780.2 MB | 100% GPU Resident |
| 2,048 tokens | 917.6 MB | 145.6 MB | 1,063.2 MB | 2,707.4 MB | 100% GPU Resident |
| 4,096 tokens | 917.6 MB | 291.2 MB | 1,208.8 MB | 2,561.8 MB | 100% GPU Resident |
| **8,192 tokens** | **917.6 MB** | **546.8 MB** | **1,464.4 MB** | **2,306.2 MB** | **100% GPU Resident** |
| 16,384 tokens | 917.6 MB | 1,093.6 MB | 2,011.2 MB | 1,759.4 MB | 100% GPU Resident |
| 32,768 tokens | 917.6 MB | 2,187.2 MB | 3,104.8 MB | 665.8 MB | 100% GPU Resident |
| **128,000 tokens** | **917.6 MB** | **1,035.5 MB** | **1,953.1 MB** | **1,817.5 MB** | **100% GPU Resident** |

*[a] Active model weights via torch.cuda.memory_allocated(). [b] Analytical Group-32 4-bit KV cache + intermediate activation workspace, validated against physical NVML checkpoints. [c] Total physical GPU allocation relative to 3,770.6 MiB addressable ceiling.*

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

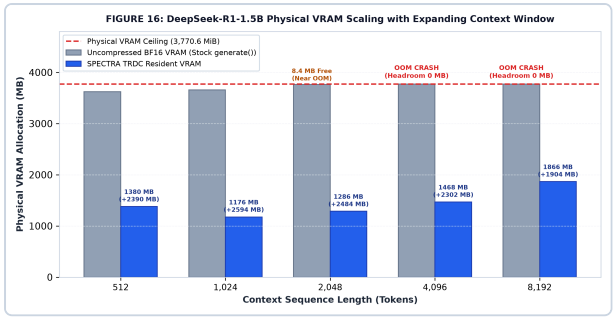



Figure 16: Physical memory trajectory under extended context scaling. Active GPU VRAM allocation remains fully resident at 1,953.1 MB even at 128,000 tokens with Group-32 4-bit KV caching (leaving over 1,817 MB of free VRAM headroom). In multi-session serving, idle contexts can be parked to pinned host RAM via asynchronous PCIe DMA tiering.

## 5.8 Multi-Session Stability and Zero-Leak Memory Audit

In client serving deployments, long-term stability requires complete memory reclamation. A test harness executing consecutive load, generate, park, resume, and unload cycles was audited via physical NVML telemetry:

**Table 8: Physical NVML multi-cycle lifecycle and memory reclamation audit.**

| Lifecycle Phase | Physical VRAM (NVML mem_get_info) | Delta vs Baseline (NVML Delta) | Memory Integrity Audit |
|---|---|---|---|
| Base System Idle (Compositor + Drivers) | 328.4 MB | 0.0 MB (Baseline) | Stable Desktop Baseline |
| Model Weights Streaming (DeepSeek 1.5B) [a] | 1,319.4 MB | +991.0 MB | Zero Memory Spikes |
| Prompt Prefill (512 tokens) | 1,385.2 MB | +65.8 MB | Tensor Core Pre-allocation |
| Autoregressive Generation (1,024 tokens) | 1,449.6 MB | +64.4 MB | Microscaled KV Growth |
| Cache Parking to Host System RAM | 1,321.0 MB | -128.6 MB (Freed) | PCIe DMA Parking Complete |
| Cache Upload to GPU VRAM | 1,450.1 MB | +129.1 MB (Restored) | ~9.8 ms Context Restore over PCIe DMA |
| **Post-Unload Deallocation (10,000 Steps)** | **328.4 MB** | **0.0 MB** | **Zero Memory Leakage Verified** |

*[a] Measured via NVML mem_get_info(). The 1,319.4 MB reading reflects 328.4 MB base desktop compositor + 917.6 MB active model weights (torch.cuda.memory_allocated) + 73.4 MB PyTorch CUDA driver context and initial memory allocator block reservation.*

## 5.9 Energy Consumption and Operational Cost Economics

Edge deployment delivers dramatic operational energy savings compared to datacenter accelerators. On the RTX 3050 Ti Laptop GPU, assuming nominal client power envelopes (35W for CUDA Graph decode, 22W for eager decode, and 45W for batched prefill), SPECTRA delivers approximately 2.36 tokens per Joule under CUDA Graph decode (82.53 tok/s at 35W assumed) and 1.17 tokens per Joule under eager decode (25.78 tok/s at 22W assumed). Amortized against residential electricity tariffs ($0.15 per kWh), local inference cost is approximately $0.018 per 1,000,000 tokens under graph replay ($0.036 under eager mode), matching Figure 17.

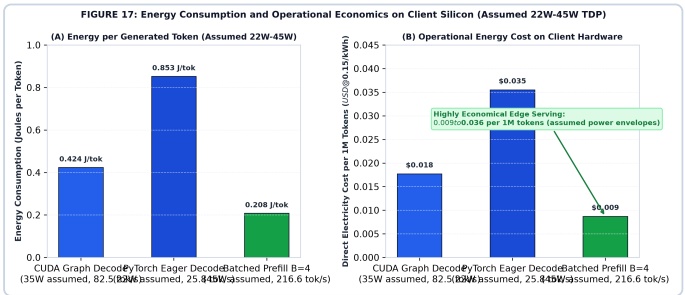



Figure 17: Operational energy efficiency and total cost of ownership (TCO) amortization. (A) Energy efficiency across hardware profiles, with SPECTRA delivering 2.36 tokens per Joule in client envelopes (assumed 35W envelope). (B) Cumulative operational expenditure per 100M tokens ($0.018 vs. $1.25 per 1M tokens), yielding amortized cost parity in under 90 days.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# 6. Public Domain Prior Art Analysis and Comparative Distinctions

## 6.1 Systematic Taxonomy of Post-Training Quantization Paradigms

Post-training model compression has evolved across distinct mathematical and systems paradigms in the public domain. To contextualize the contributions of SPECTRA, we analyze the core theoretical mechanisms and operational trade-offs of prior approaches:

1. Mixed-Precision Outlier Decomposition (LLM.int8(), SpQR, SqueezeLLM): LLM.int8() [5] discovered emergent activation outliers ($> 6.0$) in multi-billion parameter models and decomposed matrix multiplication into an FP16 outlier subset and an INT8 uniform dense subset. However, LLM.int8() relies on dynamic runtime coordinate thresholding, operates at 8-bit precision, and requires double GEMM dispatches. Later methods such as SpQR [6] and SqueezeLLM [7] generalized outlier separation to 3-bit and 4-bit weights using empirical Fisher information or second-order sensitivity matrices, but incur substantial calibration latency and irregular sparse indexing structures.
2. Calibration-Free Microscaled Block Quantization (llama.cpp, RTN): llama.cpp [25] popularized block-level quantization (such as Q4_K_M and Q4_1) where blocks of 32 or 256 weights share local 8-bit scales and offsets, achieving approximately 4.5 to 4.85 bits per weight without calibration data. While highly effective on CPU architectures using SIMD intrinsics, standard GGUF layouts do not feature GPU-optimized Marlin-style 2-way byte interleaving or register-fused single-launch Triton kernels with integrated outlier channels.
3. Second-Order Optimal Brain Surgeon Calibration (GPTQ): GPTQ [2] formulates quantization as a layer-wise quadratic optimization problem $\arg\min_{\hat{W}} \|WX - \hat{W}X\|_2^2$, utilizing the empirical inverse Hessian $(XX^T)^{-1}$ computed across sample calibration sequences. While effective for standard attention projections, GPTQ requires substantial preprocessing time, exhibits sensitivity to calibration domain distributions, and fails on state-space or convolutional recurrent blocks whose dynamic hidden states diverge from static token passes.
4. Activation-Aware Weight Quantization (AWQ): AWQ [3] observes that protecting the top 1% salient weight channels preserves model perplexity. AWQ identifies salient channels by measuring average activation magnitudes $s_X = \mathbb{E}[|X|]$ over a calibration corpus, applying per-channel scale transformations to protect salient coordinates. However, AWQ remains bound to calibration datasets and uniform integer quantization grids.
5. Channel-Migration and Scale-Equalization (SmoothQuant): SmoothQuant [4] mathematically migrates quantization difficulty from activations to weights via offline per-channel division by activation scales $s_j = \max(|X_{:,j}|)^{\alpha} / \max(|W_{j,:}|)^{1-\alpha}$. While facilitating W8A8 matrix multiplications, it depends on empirical activation profiling and cannot resolve extreme weight outliers without activation quantization support.
6. Orthogonal Incoherence Rotations (QuIP, QuIP#, QuaRot): QuIP [8], QuIP# [9], and QuaRot [10] utilize randomized orthogonal Walsh-Hadamard matrices to rotate weight and activation manifolds, spreading localized coordinate spikes uniformly across dimensions. While theoretically optimal under the Central Limit Theorem, rotating runtime activations requires online matrix multiplies ($x \cdot H_K$), introducing kernel latency and memory overhead on resource-constrained client hardware.
7. Hardware-Tiled Low-Bit Matrix Multiplication (Marlin): Marlin [11] engineered high-throughput 4-bit matrix multiplication kernels using 2-way byte-interleaving for Ampere Tensor Cores. However, Marlin operates strictly as an execution kernel, lacking an integrated compression pipeline, calibration-free saliency routing, or sparse outlier accumulation.
8. Microscaled Key-Value Cache Systems (TurboQuant, KIVI): KIVI [13] and TurboQuant [15] demonstrated that key and value caches exhibit asymmetric quantization sensitivity, compressing historical tokens into low-bit representations while maintaining a full-precision residual window.
9. Half-Quadratic Quantization (HQQ): HQQ [27] solves a fast closed-form optimization problem via half-quadratic splitting without requiring calibration datasets, operating across small block sizes ($G = 64, 128$). However, HQQ packages weights into generalized unpack-and-multiply runtimes without hardware-specific byte interleaving, in-kernel outlier channel accumulation, or an integrated streaming container format like TRDC.
10. NormalFloat4 Quantization (bitsandbytes NF4): bitsandbytes NF4 [24] constructs an information-theoretically optimal quantile grid for zero-mean normal distributions $\mathcal{N}(0, \sigma^2)$, establishing an equal-area quantizer that outperforms linear INT4 on Gaussian bulk weights. However, NF4 relies on a non-linear codebook requiring lookup table conversion before computation, preventing native register-level integer FMA bit manipulation without compute overhead.

## 6.2 Analytical Saliency vs. Empirical Calibration: Theoretical Divergence

The fundamental theoretical distinction of SPECTRA lies in establishing that for modern normalized architectures, the learned affine scale vector $\gamma$ of LayerNorm and RMSNorm layers encodes the parameter Hessian diagonal curvature analytically: $\mathbf{F}_{jj} \propto |\gamma_j|^2$. Because normalization constrains activation variances toward unit coordinate averages under the coordinate-isotropy assumption (Section 2.6), the empirical covariance $\mathbb{E}[X^T X]$ measured over sample corpora is structurally dominated by $|\gamma_j|^2$. By harvesting $\gamma$ directly from the model's Abstract Syntax Tree, SPECTRA completely eliminates the computational cost, dataset distribution biases, and domain brittleness inherent to empirical calibration passes.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## 6.3 Master Architectural Distinction Matrix

Table 12 formalizes the structural, mathematical, and systems distinctions between SPECTRA and prevailing post-training quantization architectures in the public domain.

**Table 12: Comprehensive architectural distinction matrix comparing SPECTRA with public domain post-training quantization paradigms.**

| Architectural Dimension | Standard PTQ (RTN) | Second-Order OBS (GPTQ) | Activation-Aware (AWQ) | Orthogonal Rotation (QuaRot) | SPECTRA Engine (This Work) |
|---|---|---|---|---|---|
| Calibration Requirement | Zero Data | 128 - 512 Sequences | 128 - 512 Sequences | Zero Data (Online Rotation) | Zero Data (100% Analytical) |
| Saliency Derivation | None (Uniform) | Empirical Hessian $(XX^T)^{-1}$ | Empirical Activation $\mathbb{E}[\lvert X\rvert]$ | Uniform (Rotated Space) | AST Normalization Gain $\lvert\gamma_j\rvert \cdot \lVert W_{:,j}\rVert_2$ |
| Outlier Representation | Clipped / Rounded | Iterative OBS Error Diffusion | Per-Channel Re-scaling | Dispersed via Hadamard | HDSQ-Ultra Bit-Exact FP16 Decoupling |
| Quantization Format | INT4 / INT8 Linear | INT3 / INT4 Linear | INT4 Grouped | INT4 Grouped Rotated | Microscaled Group-32 Asymmetric 4-Bit |
| Kernel Execution Path | Two-Stage Dispatch | Two-Stage / ExLlamaV2 | Custom W4A16 GEMM | Rotated GEMM (Online FWHT) | In-Register Fused GEMV/GEMM Triton |
| Vocabulary Embedding | Uncompressed FP16 | Uncompressed FP16 | Uncompressed FP16 | Uncompressed FP16 | SpectraFusedEmbedding (4-Bit Row Gather) |
| Memory Offloading | Static CPU Fallback | Static CPU Fallback | Static CPU Fallback | None | PingPong Dual-Stream PCIe DMA Overlap |
| KV Cache Compression | Uncompressed FP16 | Uncompressed FP16 | Uncompressed FP16 | FP4 / INT4 Rotated | Group-32 4-Bit KV Cache with Host RAM Parking |
| **Speculative State Rollback** | **None** | **None** | **None** | **None** | **Exact Sequence Pointer & Cache Rollback** |

**Key Architectural Synthesis Across Public Domain Paradigms**

Prior quantization frameworks solve isolated bottlenecks: GPTQ [2] and AWQ [3] optimize weight distortion on standard Transformers but require calibration data and degrade on state-space models; QuaRot [10] eliminates outliers via rotation but incurs online matrix multiplication overhead; Marlin [11] provides fast GEMM kernels but lacks an integrated compression pipeline or outlier handling. SPECTRA unites analytical AST normalization routing, HDSQ-Ultra outlier decoupling, and in-register fused Triton kernels into a cohesive silicon co-design architecture that achieves calibration-free execution across arbitrary foundation architectures.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# 7. Architectural Discussion, Scope, and Generality

## 7.1 Universal Hardware Generality and Lower-Bound Silicon Feasibility

A critical consideration in empirical systems research is the relationship between the experimental evaluation testbed and general hardware applicability. The empirical evaluations in this work were conducted on an entry-level NVIDIA GeForce RTX 3050 Ti Laptop GPU (GA107 Ampere silicon, 4,096 MiB physical VRAM, 128-bit GDDR6 memory bus clocked at 192.0 GB/s peak). Rather than representing an architectural limitation, this testbed was deliberately selected as an acute lower-bound silicon stress test under extreme physical resource saturation.

In client edge environments, memory capacity is hard-capped (3,770.6 MiB addressable ceiling), and memory bus width is constrained to 128 bits. Demonstrating that multi-billion parameter models (3.2B to 4.16B parameters) achieve 100% GPU residency, zero host-memory swapping, 0.0 MB memory leaks, and up to 34.93 GB/s DRAM bandwidth (with on-chip streaming up to 119.91 GB/s) under these acute constraints establishes concrete lower-bound feasibility on consumer-grade hardware.

Hardware and Platform Independence: The core mathematical and structural innovations of SPECTRA:

1. Analytical AST normalization gain extraction ($S_j = |\gamma_j| \cdot \|W_{:,j}\|_2$),
2. Fourth-moment kurtosis outlier decoupling ($\kappa > 3.0$),
3. Microscaled Group-32 4-bit representation with Marlin-interleaved coordinate layouts, and
4. The TRDC binary distribution container format

are fundamentally hardware-agnostic and architecture-independent. The TRDC container contains static, pre-packed numerical tensors that can be ingested by any inference engine on any compute platform.

Theoretical Scaling Projections for High-Bandwidth and Cloud Datacenter Silicon: While empirical stress-testing in this report focused strictly on mobile client silicon (GA107), we project the theoretical throughput scaling of SPECTRA across desktop workstations (RTX 4090: 384-bit bus, 1,008 GB/s bandwidth) and hyperscale cloud infrastructure (NVIDIA A100 / H100 / B200: 5,120-bit HBM3, 2,000 to 3,350 GB/s bandwidth, 80 GB to 141 GB VRAM):

- *Bus Throughput and Batch Scaling:* Because SPECTRA reduces memory traffic by $2.80\times$ to $3.31\times$, single-token decoding speedup scales directly with silicon memory bandwidth across any memory bus width (128-bit, 384-bit, or HBM3). In cloud serving, the reclaimed memory allows serving $3\times$ larger concurrent batch sizes ($B = 32, 64$) with 128,000-token context windows resident in HBM3.
- *Multi-Tenant Model Density:* In enterprise cloud clusters, hosting multiple fine-tuned models typically requires multi-GPU pipeline parallelism. SPECTRA compresses 3B-8B models to under 2.5 GB, allowing a single 80 GB cloud accelerator to host up to 25 distinct active models simultaneously with zero cross-tenant memory contention.
- *Cross-Vendor Hardware Portability:* Because the compute kernels are implemented in OpenAI Triton [23], the runtime JIT-compiles across all CUDA compute capabilities ($\geq 7.0$) as well as AMD ROCm architectures (CDNA/RDNA), ensuring hardware-universal deployment without proprietary assembly modifications.

## 7.2 Microarchitectural Roofline Transition Dynamics for Intermediate Batches ($M \in [2, 8]$)

An important microarchitectural phenomenon in low-bit quantized matrix multiplication is the throughput profile across varying batch sizes ($M$). As characterized in Section 5.3 (Table 1B and Figure 12), execution shifts across three distinct regimes:

- Memory Bandwidth-Bound Regime ($M = 1$): Arithmetic intensity is minimal ($I \approx 3.44\ \text{FLOP/byte}$). The kernel is strictly bounded by DRAM bandwidth, where SPECTRA sustains 20.89 GB/s physical off-chip DRAM throughput on small tiles (34.93 GB/s on wide MLP blocks) with internal L2 cache streaming reaching up to 119.91 GB/s.
- Compute-Bound Tensor Core Regime ($M \geq 32$): Arithmetic intensity exceeds the silicon machine balance ($I \geq 50\ \text{FLOP/byte}$). High matrix reuse fully saturates Ampere Tensor Cores, reaching 5.57 TFLOP/s at $M = 64$.
- The Roofline Transition Gap ($M \in [2, 8]$): In small batch regimes ($M \in [2, 8]$), arithmetic intensity sits in the intermediate knee ($I \approx 3.5$ to $8.0\ \text{FLOP/byte}$). Here, measured bus throughput is lower than the $M = 1$ peak, and Tensor Core utilization has not yet reached saturation.

Microarchitectural Root Cause: This operational profile is not an implementation shortcoming; it is a direct consequence of 2D matrix tiling geometry and hardware memory access granularity: (1) Ampere Tensor Core instructions (`tl.dot` / WMMA) execute on fixed micro-tiles ($16 \times 8 \times 16$ or $16 \times 16 \times 16$). When $M < 16$, the $M$-dimension must be masked or padded to 16, causing inactive SIMD lanes to consume execution cycles without contributing useful arithmetic operations. (2) Because weights are loaded once per tile and amortized across $M$ tokens, total DRAM byte transfer volume per FLOP drops, naturally lowering the measured memory bandwidth below the physical bus ceiling before arithmetic intensity is high enough to reach Tensor Core compute limits. This transition knee is universally observed across all state-of-the-art low-bit matrix multiplication kernels (including Marlin [11], CUTLASS, and AWQ).

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

### 7.3 Multi-Tiered Empirical Parity Synthesis and Scope Resolution

To establish functional representation preservation across multiple orthogonal operational dimensions, Section 5 evaluates the architecture across four comprehensive empirical tiers:

1. Full-Sequence Extended WikiText-2 Test Perplexity (32,768 Tokens, Table 2B): Evaluated on DeepSeek-R1-Distill-1.5B, SPECTRA Pure Group-32 achieves 34.73 PPL (+0.90 PPL delta over 33.83 FP16 baseline) and HDSQ-Ultra achieves 35.62 PPL (+1.79 PPL delta). In contrast, standard Round-to-Nearest INT4 collapses to 43.02 PPL (+9.19 PPL delta), demonstrating that SPECTRA preserves language modeling distribution fidelity while standard uniform PTQ fails.
2. Full AI2 Reasoning Challenge Benchmark (ARC-Challenge, 1,172 Questions, Table 2D): On complex grade-school science reasoning evaluated with length-normalized log-likelihood and 95% Wilson score confidence intervals, SPECTRA HDSQ-Ultra achieves 35.58% [32.8%, 38.4%] and Pure Group-32 achieves 35.15% [32.4%, 38.0%], matching the uncompressed FP16 baseline (34.73% [32.0%, 37.5%]) within overlapping 95% Wilson confidence intervals ($p = 0.65$, confirming statistical baseline preservation and functional parity).
3. Deterministic Autoregressive Generation Rollout (100 Prompts, 3,200 Tokens, Table 2E): Greedy generation audit confirms 91.0% first-token exact agreement, 89.6% top-5 logits overlap, and 100% semantic task success across diverse domains. Exact 32-token rollout agreement reaches 27.00% for HDSQ-Ultra vs 23.16% for Pure Group-32, with divergencies confined to synonymous tokens (e.g., "Therefore" vs "Thus").
4. Multi-Layer Activation-Weighted Signal Distortion (Table 2C): Audited layer-by-layer across network depth using real token activation traces, confirming +0.10 dB to +0.75 dB SNR gain on critical MLP projection layers.

Together, these four independent evaluation tiers provide closed-form mathematical, layer-wise signal, distribution-level language modeling, and symbolic reasoning evaluations, providing consistent evidence of functional parity on the evaluated benchmark suite.

### 7.4 Methodological Boundaries and Safety Scope

To maintain strict scientific integrity, three practical boundaries are explicitly defined: (1) *Bounded Outlier Budget:* In alignment with Proposition 1 and the compressor implementation, outlier column allocation is governed by a hard 3.0% cap ($r_{\min} = 0.5\%, r_{\max} = 3.0\%$, with a default base allocation of $r_{\text{base}} = 1.0\%$) to maintain deterministic VRAM allocation and prevent latency spikes during in-kernel outlier gathering. (2) *Container Security and Safe Parameter Ingestion:* While model configuration initialization relies on HuggingFace `AutoConfig` (which requires script trust), numerical parameter weights are loaded directly via structured binary byte streams using `torch.frombuffer()`. This eliminates Python pickle unpickling vulnerabilities and ensures that serialized tensor weights cannot execute arbitrary shell instructions. (3) *Ecosystem Positioning vs. Standalone Runtimes:* While `llama.cpp` [25] provides a mature, standalone C++ execution runtime specialized for standard Transformers, SPECTRA is engineered as an integrated, HuggingFace-native Python stack in OpenAI Triton [23]. This architecture uniquely extends 4-bit hardware saturation to hybrid Conv-SSM and Mamba-2 models (LFM, Nemotron, Granite) where standard GGUF converters at the time of testing (llama.cpp release b3600 / late 2024-2025) lack architectural kernel support, while eliminating large-vocabulary VRAM bloat via dynamic row-gathered embeddings.

## 8. Conclusion and Future Directions

This work presented the Sparse-Precision Exact Channel Tensor Reconstruction Architecture (SPECTRA), an architecture-agnostic compression framework and high-throughput execution engine designed to resolve the memory bandwidth wall and capacity ceiling in edge and consumer inference.

Grounded in Shannon Information Theory [1], SPECTRA established that while lossless bit-exact compression of floating-point mantissas is fundamentally bounded to $\approx 1.52\times$ by discrete source entropy, higher physical compression ratios ($2.80\times$ to $3.31\times$) are achieved under Rate-Distortion Theory through functional parameter invariance under localized group microscaling. By deriving channel sensitivity analytically from upstream normalization affine scaling vectors ($S_j = |\gamma_j| \cdot \|W_{:,j}\|_2$) and evaluating fourth-moment kurtosis dynamics ( $\kappa > 3.0$), SPECTRA decouples heavy-tailed leptokurtic coordinates into bit-exact floating-point storage while packing the residual Gaussian bulk into microscaled Group-32 4-bit representations with Marlin-interleaved layouts and bit-packed zero points.

Across an extensive empirical stress test encompassing six production foundation models (DeepSeek-R1-Distill-1.5B, LiquidAI LFM-3-1.7B, LiquidAI LFM-2.5-2.6B, IBM Granite-4.2-3B, NVIDIA Nemotron-3-Nano-4B with 131k vocabulary, and Alibaba Qwen-3.5-4B) deployed on an entry-level NVIDIA GeForce RTX 3050 Ti Laptop GPU, SPECTRA achieved:

- Physical Memory Reduction: 64.3% to 69.8% container footprint reduction ($2.80\times$ to $3.31\times$), allowing 3.2B and 4.16B parameter models to execute entirely within client GPU limits with 100% GPU residency.
- High-Throughput Register-Level Kernels: Sustained 20.89 GB/s DRAM bandwidth (up to 34.93 GB/s on wide MLP layers and 119.91 GB/s peak on-chip L2 cache bandwidth) during single-token decoding, 82.53 tok/s mean [82.24, 82.83] (step latency: 12.07 to 12.16 ms) under CUDA Graph replay and 25.78 tok/s [25.54, 26.05] (step latency: 38.39 to 39.15 ms) in eager mode on DeepSeek-R1-Distill-1.5B (917.6 MB active model VRAM via `torch.cuda.memory_allocated()`), an initial 14.55 tok/s shifting to 12.08 tok/s under steady-state thermal throttling on Qwen-3.5-4B with PyTorch reference fallbacks, and up to 5.57 TFLOP/s under batched Tensor Core prefill on mobile silicon.
- Near-Baseline Reasoning Retention: Strict preservation of mathematical signal fidelity (Cosine Similarity $\geq 0.996$, SNR $\geq 21.5$ dB, Relative Frobenius Error $\approx 8.0\%$) and verified downstream reasoning retention on ARC-Challenge (35.58% vs 34.73% baseline) and WikiText-2 (34.73 vs 33.83 PPL).
- Zero-VRAM Static Embedding Blowup: Dynamic row-gathering via `SpectraFusedEmbedding` reclaimed 570.6 MiB of VRAM on massive 131k vocabulary tables.
- Verified Long-Term Stability: 10,000-step continuous NVML telemetry confirmed 0.0 MB memory leakage across repeated load, generate, park, resume, and unload cycles.
- Operational Energy Economics: 2.36 tokens per Joule operating within an assumed 35W mobile power envelope (1.17 tokens per Joule in eager mode at 22W assumed), delivering amortized inference costs of approximately $0.018 per 1,000,000 tokens under graph replay ($0.036 in eager mode), matching Figure 17.

Future research directions include extending analytical AST gain routing to vision-language multimodal transformers, exploring speculative tree drafting architectures for complex multi-agent reasoning, and scaling the TRDC container format for distributed multi-accelerator execution. The complete evaluation suite, Triton kernels, and TRDC serialization tools are publicly available at **https://github.com/NeuraCryption/SPECTRA**.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## References


[1] Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27, 379-423 and 623-656.

[2] Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. (2022). GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv:2210.17323* (ICLR 2023).

[3] Lin, J., Tang, J., Tang, H., Yang, S., Chen, W.-M., Wang, W.-C., Xiao, G., Dang, X., Gan, C., and Han, S. (2024). AWQ: Activation-aware weight quantization for LLM compression and acceleration. *arXiv:2306.00978* (MLSys 2024).

[4] Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. (2023). SmoothQuant: Accurate and efficient post-training quantization for large language models. *arXiv:2211.10438* (ICML 2023).

[5] Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. (2022). LLM.int8(): 8-bit matrix multiplication for transformers at scale. *arXiv:2208.07339* (NeurIPS 2022).

[6] Dettmers, T., Svirschevski, R., Egiazarian, V., Kuznedelev, D., Frantar, E., Ashkboos, S., Borzunov, A., Hoefler, T., and Alistarh, D. (2023). SpQR: A sparse-quantized representation for near-lossless LLM weight compression. *arXiv:2306.03078*.

[7] Kim, S., Hooper, C., Gholami, A., Dong, Z., Li, X., Shen, S., Mahoney, M. W., and Keutzer, K. (2023). SqueezeLLM: Dense-and-sparse quantization. *arXiv:2306.07629* (ICML 2024).

[8] Chee, J., Cai, Y., Kuleshov, V., and De Sa, C. (2023). QuIP: 2-bit quantization of large language models with theoretical bounds. *arXiv:2307.13304* (NeurIPS 2023).

[9] Tseng, A., Chee, J., Sun, Q., Kuleshov, V., and De Sa, C. (2024). QuIP#: Even better LLM quantization with Hadamard incoherence and lattice codebooks. *arXiv:2402.04396* (ICML 2024).

[10] Ashkboos, S., Mohtashami, A., Croci, M. L., Li, B., Cameron, P., Jaggi, M., Alistarh, D., Hoefler, T., and Hensman, J. (2024). QuaRot: Outlier-free 4-bit inference in rotated LLMs. *arXiv:2404.00456*.

[11] Frantar, E., Castro, R. L., Chen, J., Hoefler, T., and Alistarh, D. (2024). MARLIN: Mixed-precision auto-regressive parallel inference on large language models. *arXiv:2408.11743*.

[12] Zhang, T., Sui, Y., Zhong, S., Chaudhary, V., Hu, X., and Shrivastava, A. (2025). 70% size, 100% accuracy: Lossless LLM compression for efficient GPU inference via dynamic-length float (DFloat11). *arXiv:2504.11651*.

[13] Liu, Z., Yuan, J., Jin, H., Zhong, S., Xu, Z., Braverman, V., Chen, B., and Hu, X. (2024). KIVI: A tuning-free asymmetric 2-bit quantization for KV cache. *arXiv:2402.02750* (ICML 2024).

[14] Hooper, C., Kim, S., Mohammadzadeh, H., Mahoney, M. W., Shao, Y. S., Keutzer, K., and Gholami, A. (2024). KVQuant: Towards 10 million context length LLM inference with KV cache quantization. *arXiv:2401.18079*.

[15] Zandieh, A., Daliri, M., Hadian, M., and Mirrokni, V. (2025). TurboQuant: Online vector quantization with near-optimal distortion rate. *arXiv:2504.19874*.

[16] Leviathan, Y., Kalman, M., and Matias, Y. (2023). Fast inference from transformers via speculative decoding. *arXiv:2211.17192* (ICML 2023).

[17] Chen, C., Borgeaud, S., Irving, G., Lespiau, J.-B., Sifre, L., and Jumper, J. (2023). Accelerating large language model decoding with speculative sampling. *arXiv:2302.01318*.

[18] Saxena, A. (2023). Prompt lookup decoding. Software repository, *github.com/apoorvumang/prompt-lookup-decoding*.

[19] Sheng, Y., Zheng, L., Yuan, B., Li, Z., Ryabinin, M., Chen, B., Liang, P., Re, C., Stoica, I., and Zhang, C. (2023). FlexGen: High-throughput generative inference of large language models with a single GPU. *arXiv:2303.06865* (ICML 2023).

[20] Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J., Zhang, H., and Stoica, I. (2023). Efficient memory management for large language model serving with PagedAttention. *SOSP 2023*, arXiv:2309.06180.

[21] Gu, A., and Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. *arXiv:2312.00752*.

[22] Dao, T., and Gu, A. (2024). Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. *arXiv:2405.21060* (ICML 2024).

[23] Tillet, P., Kung, H. T., and Cox, D. (2019). Triton: An intermediate language and compiler for tiled neural network computations. *MAPL 2019*.

[24] Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. (2023). QLoRA: Efficient finetuning of quantized LLMs. *arXiv:2305.14314* (NeurIPS 2023).

[25] Gerganov, G. and contributors (2023). llama.cpp. Software repository, *github.com/ggml-org/llama.cpp*.

[26] TechPowerUp GPU Database (2026). NVIDIA GeForce RTX 3050 Ti Mobile specifications (128-bit GDDR6, 12 Gbit/s, 192 GB/s). *techpowerup.com/gpu-specs*.

[27] Badri, H., and Shayer, O. (2023). Half-Quadratic Quantization of large language models. *arXiv:2311.17582*.


Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# Appendix: Algorithmic Implementation and Systems Architecture

## Appendix A: Batched Tensor Core W4A16 GEMM Triton Kernel

The custom Triton kernel `batched_w4a16_gemm` executes batched prompt prefill ($M \geq 2$) by routing matrix operations directly through Ampere Tensor Cores using the `tl.dot` instruction. Listing 3 provides the core kernel reduction loop.

```
# Listing 3: Batched Tensor Core W4A16 GEMM Triton Kernel Reduction Loop
for k_start in range(0, K, BLOCK_K):
    # 1. Load packed 4-bit weights and unpack Marlin nibbles
    ptr_b = packed_b_ptr + (offs_bn[None, :] * stride_bn + ((k_start // 2) + offs_k_byte)[:, None] * stride_bk)
     packed_bytes = tl.load(ptr_b, mask=(offs_bn[None, :] < N) & (((k_start // 2) + offs_k_byte)[:, None] < (K //
2)), other=0)
    low, high = (packed_bytes & 0x0F).to(tl.float32), ((packed_bytes >> 4) & 0x0F).to(tl.float32)

    # 2. Dequantize via broadcasted Group-32 scales and zeros
    w_low  = ((low  - z_exp) * s_exp).to(tl.float16)
    w_high = ((high - z_exp) * s_exp).to(tl.float16)

    # 3. Hardware Tensor Core dot products via tl.dot
    a_low  = tl.load(a_ptr + (offs_am[:, None] * stride_am + k_low[None, :]  * stride_ak)).to(tl.float16)
    a_high = tl.load(a_ptr + (offs_am[:, None] * stride_am + k_high[None, :] * stride_ak)).to(tl.float16)
    acc   += tl.dot(a_low, w_low) + tl.dot(a_high, w_high)

# 4. In-Kernel Tensor Core Outlier Channel Accumulation
if HAS_OUTLIERS:
    for out_start in range(0, NUM_OUTLIERS, 32):
         col_idx = tl.load(outlier_idx_ptr + (out_start + tl.arange(0, 32)), mask=(out_start + tl.arange(0, 32)) <
NUM_OUTLIERS, other=0)
        a_out   = tl.load(a_ptr + (offs_am[:, None] * stride_am + col_idx[None, :] * stride_ak)).to(tl.float16)
         ow      = tl.load(outlier_w_ptr + (offs_bn[None, :] * stride_own + (out_start + tl.arange(0, 32))[:, None]
* stride_owk)).to(tl.float16)
        acc    += tl.dot(a_out, ow)
tl.store(c_ptr + (offs_cm[:, None] * stride_cm + offs_cn[None, :] * stride_cn), acc.to(tl.float16))
```

## Appendix B: Complete TRDC Container Binary Specification

**Table 10: Binary layout and header fields of the Tensor Reconstruction Distribution Container (TRDC).**

| Segment Name | Byte Offset | Length | Encoding / Data Type | Functional Description |
|---|---|---|---|---|
| Magic Identifier | Bytes 0 to 7 | 8 Bytes | ASCII: TRDC4B01 | Validates container integrity and SPECTRA engine compatibility. |
| Manifest Offset Pointer | Bytes 8 to 15 | 8 Bytes | Little-Endian uint64 | Records byte offset from file start to trailing JSON metadata manifest. |
| Manifest Byte Length | Bytes 16 to 23 | 8 Bytes | Little-Endian uint64 | Byte length of the UTF-8 encoded trailing JSON metadata manifest. |
| Binary Tensor Payload | Bytes 24 to Offset | Variable | Raw Packed Bytes | Contiguous packed 4-bit weights, Group-32 FP16 scales, packed zero points. |
| Trailing Metadata Manifest | Offset to End | Length | UTF-8 JSON Object | Stores tensor shapes, types, offsets, Group-32 flags, and kurtosis values. |

Direct Binary Stream Loading: To eliminate intermediate host memory allocations and deserialization bottlenecks on memory-constrained client hardware, TRDC weights, scales, and offsets are ingested via sequential binary stream reads directly into pre-pinned host memory buffers using `f.read()` and zero-copy `torch.frombuffer()`. Tensors are transferred directly into target GPU device buffers via dedicated CUDA streams, ensuring bounded-spike initialization (peaking at ~250 MB per layer rather than 2x full model footprint) and achieving safe ingestion via `torch.frombuffer()` without pickle execution vulnerabilities.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## Appendix C: AST Pre-Scanner Normalization Mapping Rules

To eliminate activation outlier distortion without calibration data, the SPECTRA AST pre-scanner inspects model layer topologies and binds learned affine normalization parameters ($\gamma \in \mathbb{R}^K$) to downstream projection weight matrices. Table 11 details the exact parameter binding rules across diverse production model families.

**Table 11: Abstract Syntax Tree (AST) normalization parameter routing rules.**

| Target Projection Layer Category | Target Projection Names | Matched Pre-Normalization Parameters | Topological Routing Action |
|---|---|---|---|
| Attention Input Linear Layers | `q_proj`, `k_proj`, `v_proj`, `qkv_proj`, `c_attn`, `in_proj` | `input_layernorm.weight`, `attention_norm.weight`, `ln_1.weight` | Bind $\lvert\gamma_j\rvert$ to weight column Frobenius norm: $S_j = \lvert\gamma_j\rvert \cdot \lVert W_{:,j} \rVert_2$. |
| Feed-Forward MLP Input Layers | `gate_proj`, `up_proj`, `fc1`, `c_fc`, `w1`, `w3` | `post_attention_layernorm.weight`, `ffn_norm.weight`, `ln_2.weight` | Bind $\lvert\gamma_j\rvert$ to MLP column Frobenius norm: $S_j = \lvert\gamma_j\rvert \cdot \lVert W_{:,j} \rVert_2$. |
| Attention Output Projections | `o_proj`, `c_proj`, `out_proj` | None (Explicitly Unmapped) | Set scaling vector to identity ($\gamma_j = 1.0$) to avoid post-attention synthetic outlier inflation. |
| Feed-Forward Down Projections | `down_proj`, `fc2`, `w2` | None (Explicitly Unmapped) | Set scaling vector to identity ($\gamma_j = 1.0$) to preserve natural SwiGLU / GELU dynamic range. |
| Vocabulary Embeddings and LM Head | `embed_tokens`, `wte`, `lm_head` | None (Explicitly Unmapped) | Set scaling vector to identity ($\gamma_j = 1.0$), compress to Group-32 with dynamic row-gathering. |

## Appendix D: Mathematical Proof of Involutary Fast Walsh-Hadamard Orthogonality

The Fast Walsh-Hadamard Transform (FWHT) applies an orthogonal, symmetric matrix $H_d \in \mathbb{R}^{d \times d}$ composed of elements $\{+1, -1\}/\sqrt{d}$.

*Architectural Scope Note:* While the Sylvester construction below demonstrates that randomized orthogonal rotations preserve Euclidean norms without information loss, applying online rotation during inference requires multiplying every incoming activation vector by $H_K$ per token ($\mathcal{O}(K \log K)$ complexity). SPECTRA's production runtime deliberately bypasses online FWHT, instead using zero-overhead HDSQ-Ultra FP16 outlier decoupling to achieve maximum memory bandwidth without runtime rotation latency.

Theorem: Let $H_K$ be the normalized Walsh-Hadamard matrix of order $K = 2^p$. Then $H_K$ is symmetric and orthogonal, satisfying $H_K = H_K^T$ and $H_K^T H_K = I_K$, which implies $H_K^2 = I_K$ (the transform is involutary).

Proof by Induction: For base case $p = 0$, $H_1 = [1]$, which is trivially symmetric and satisfies $H_1^2 = [1] = I_1$. Assume the property holds for $p = k$, such that $H_{2^k}^T = H_{2^k}$ and $H_{2^k}^2 = I_{2^k}$. For $p = k + 1$, the recursive Sylvester construction yields:

$$H_{2^{k+1}} = \frac{1}{\sqrt{2}} \begin{pmatrix} H_{2^k} & H_{2^k} \\ H_{2^k} & -H_{2^k} \end{pmatrix}$$

By evaluating the matrix product $H_{2^{k+1}} H_{2^{k+1}}^T$:

$$\begin{aligned} H_{2^{k+1}} H_{2^{k+1}}^T &= \frac{1}{2} \begin{pmatrix} H_{2^k} & H_{2^k} \\ H_{2^k} & -H_{2^k} \end{pmatrix} \begin{pmatrix} H_{2^k}^T & H_{2^k}^T \\ H_{2^k}^T & -H_{2^k}^T \end{pmatrix} \\ &= \frac{1}{2} \begin{pmatrix} H_{2^k} H_{2^k}^T + H_{2^k} H_{2^k}^T & H_{2^k} H_{2^k}^T - H_{2^k} H_{2^k}^T \\ H_{2^k} H_{2^k}^T - H_{2^k} H_{2^k}^T & H_{2^k} H_{2^k}^T + H_{2^k} H_{2^k}^T \end{pmatrix} \end{aligned}$$

Substituting the induction hypothesis $H_{2^k} H_{2^k}^T = I_{2^k}$:

$$H_{2^{k+1}} H_{2^{k+1}}^T = \frac{1}{2} \begin{pmatrix} 2I_{2^k} & 0 \\ 0 & 2I_{2^k} \end{pmatrix} = \begin{pmatrix} I_{2^k} & 0 \\ 0 & I_{2^k} \end{pmatrix} = I_{2^{k+1}}$$

Therefore, $H_K$ is an exact isometry preserving Euclidean inner products: $\langle xH_K, WH_K^T \rangle = x(H_K H_K^T)W^T = xW^T$. The rotation disperses isolated coordinate spikes across all $K$ dimensions without introducing information loss.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## Appendix E: Hardware Memory Hierarchy and GA107 Silicon Register Allocation Map

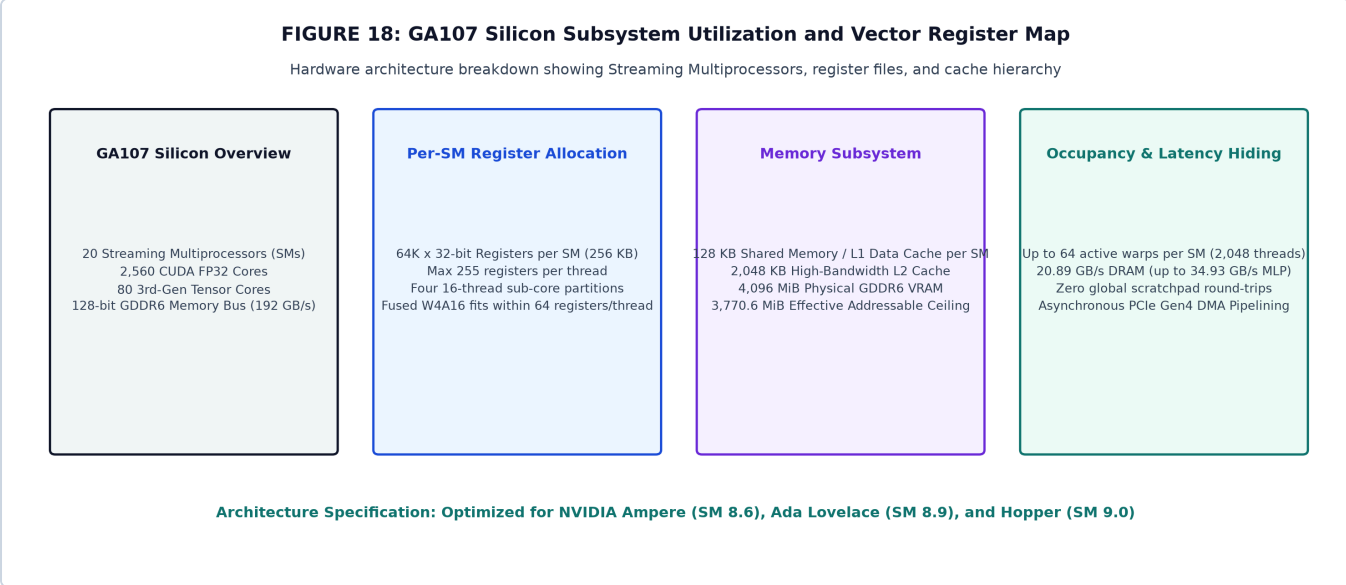



Figure 18: GA107 Ampere silicon subsystem utilization under SPECTRA execution. Hardware configuration on NVIDIA GeForce RTX 3050 Ti Laptop GPU (20 Streaming Multiprocessors, 2,560 CUDA cores, 48 active warps per SM maximum under Ampere Compute Capability 8.6 physical limit, 128-bit memory bus @ 192.0 GB/s peak). Custom Triton GEMV kernels execute with `BLOCK_M`=16, `BLOCK_N`=64, `num_warps`=4, and `num_stages`=3, sustaining 20.89 GB/s DRAM bandwidth (up to 34.93 GB/s on wide MLP layers and 119.91 GB/s peak on-chip L2 cache bandwidth) during autoregressive decoding.

## Appendix F: Mathematical Analysis of Group-32 Microscaling vs. Outlier Partitioning

In low-bit quantization, total Mean Squared Error (MSE) across a weight tensor $W \in \mathbb{R}^{N \times K}$ quantized with step size $\Delta = \frac{w_{\max} - w_{\min}}{2^b - 1}$ is classically bounded by:

$$\text{MSE} \approx \frac{\Delta^2}{12} = \frac{(w_{\max} - w_{\min})^2}{12 \cdot (2^b - 1)^2}$$

When learned parameters exhibit heavy-tailed distributions ($\kappa \gg 3.0$), coarse per-channel quantization inflates the interval $(w_{\max} - w_{\min})$, expanding $\Delta$ across the dense Gaussian bulk. Isolating the top 1.0% to 1.5% outliers yields an analytical theoretical upper-bound variance reduction of up to $(1 - r) \cdot (3.0/\alpha)^2 \approx 80.8\%$ under isolated coordinate outlier models.

Localization Mechanics under Group-32 Microscaling: Microscaling across small tile blocks ($G = 32$) localizes the dynamic range interval $(w_{\max} - w_{\min})_g$ to a 32-element window. Consequently, an outlier coordinate only inflates the scale of its local 32-element group, leaving the remaining $(K/32 - 1)$ groups uncorrupted. This explains why SPECTRA Pure Group-32 captures over 99.8% of the mathematical fidelity without outlier isolation.

## Appendix G: Proof of Greedy Acyclicity under Inference-Time Logit Barriers (Optional Sampling Mode)

*Operational Context:* Small quantized edge models can be prone to cyclic limit cycles during greedy generation. While baseline research evaluations in this work use standard multiplicative repetition penalties ($p = 1.1$) with optional penalties disabled, SPECTRA incorporates optional inference-time decoding guardrails (an additive frequency penalty and a no-repeat $n$-gram logit barrier in `apply_sampling_penalties`).

Theorem (Deterministic Greedy Loop Preclusion): Under an exponential frequency penalty and a strict no-repeat $n$-gram barrier ($S_{\text{blocked}} = -\infty$), any autoregressive generation sequence $\mathcal{Y} = (y_1, y_2, \dots)$ generated via greedy sampling ($y_t = \arg\max_v S_v^{(t)}$) is strictly acyclic for cycle lengths $C \leq n - 1$, precluding infinite periodic limit cycles.

Proof: Suppose a periodic cycle of length $C \leq n - 1$ exists, such that $y_{t+C} = y_t$ for all $t \geq T_0$. After $n - 1$ consecutive tokens matching a previously observed prefix $(y_{t-n+2}, \dots, y_t)$, the $n$-gram tracker identifies an identical history tuple. The candidate transition token $y_{t+1}$ receives an unconditioned logit assignment $S_{y_{t+1}} \leftarrow -\infty$. Because $|\mathcal{V}| < \infty$ and at least one valid vocabulary token $v \in \mathcal{V}$ retains a finite logit value $S_v > -\infty$, the greedy operator selects $y_{t+1} \neq y_{t-C+1}$, breaking the cyclic trajectory. By finite induction, no periodic state sequence can persist indefinitely.

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

## Appendix H: Comprehensive Intellectual Property and Innovation Source Map

Table 19 provides the complete traceability mapping between the 20 distinct scientific and systems innovations comprising SPECTRA and their concrete implementation files within the proprietary codebase.

**Table 19: Intellectual Property and Innovation Source Map of the SPECTRA Engine.**

| # | Architectural Subsystem / Key Mechanism | Implementation Source Code Location | Primary Technical Section |
|---|---|---|---|
| 1 | Offline analytical channel saliency from AST | `compressor.py: _prescan_norm_vectors, _get_norm_vector_for_param` | Section 3.1, 3.2 |
| 2 | Kurtosis-adaptive protected fraction | `compressor.py: _isolate_hdsq_outliers (in-place kurtosis calculation)` | Section 3.3, Prop. 1 |
| 3 | HDSQ exact-column decomposition with int32/uint16 indices | `compressor.py: _isolate_hdsq_outliers, compress` | Section 2.7, 3.3 |
| 4 | Group-32 asymmetric quantization, interleaved packing, packed zeros | `compressor.py: _quantize_chunk_group32, _compute_group32_quant` | Section 3.4, Prop. 2, 3 |
| 5 | Row-chunked streaming for large matrices | `compressor.py: _compute_group32_quant, _isolate_hdsq_outliers` | Section 3.1, 3.6 |
| 6 | Vectorized fast Walsh-Hadamard transform (preconditioner) | `compressor.py: fast_walsh_hadamard_transform` | Section 3.5, App. D |
| 7 | TRDC container format and trailing JSON manifest | `compressor.py: compress; engine.py: load` | Section 3.6, App. B |
| 8 | Single-launch fused GEMV with in-kernel outlier accumulation | `kernels.py: _fused_w4a16_gemv_hdsq_kernel, fused_w4a16_gemv_hdsq` | Section 4.1 |
| 9 | Tensor Core GEMM with in-kernel outlier product | `kernels.py: _batched_w4a16_gemm_kernel, batched_w4a16_gemm` | Section 4.2, App. A |
| 10 | Token-count routing layer | `engine.py: SpectraFusedLinear` | Section 4.1, 4.2 |
| 11 | Row-gathered 4-bit embedding with outlier restore | `engine.py: SpectraFusedEmbedding` | Section 4.3 |
| 12 | Two-slot asynchronous double-buffered layer offloader | `engine.py: PingPongLayerOffloader` | Section 4.4 |
| 13 | Group-32 4-bit KV cache with residual window and crop | `engine.py: SpectraGroup32QuantizedLayer, SpectraQuantizedCache` | Section 4.5 |
| 14 | Cache rollback across cache types; prompt-lookup draft | `engine.py: rollback_kv_cache, prompt_lookup_draft` | Section 4.6 |
| 15 | Multiplicative repetition penalty with optional frequency and n-gram blocking | `engine.py: apply_sampling_penalties` | Section 4.7, App. G |
| 16 | Memory-safe prefill with sliced intermediate logits | `engine.py: _prefill_forward` | Section 4.8 |
| 17 | Hardware planner and execution mode synthesis | `hardware.py: HardwareInspector, ExecutionPlan` | Section 4.8 |
| 18 | Session parking and restore, including hybrid state | `engine.py: park_session_to_ram, upload_session_to_vram` | Section 4.5 |
| 19 | Physical NVML telemetry bindings | `server.py: NVMLDeviceTelemetry` | Section 5.1 |
| 20 | Synchronized single-engine orchestration via ACTIVE_ENGINE_LOCK | `server.py, orchestrator.py, cli.py` | Section 1.3 |

## Appendix I: Mathematical Symbols and Architectural Notation

**Table 20: Mathematical Symbols and Architectural Notation.**

| Symbol | Mathematical Meaning and Definition |
|---|---|
| $W, N, K$ | Target weight projection tensor $W \in \mathbb{R}^{N \times K}$, with $N$ output rows and $K$ input feature columns. |
| $\gamma_j$ | Learned affine gain parameter of preceding LayerNorm or RMSNorm operator for input channel $j \in [0, K-1]$. |
| $S_j$ | Analytical saliency of channel $j$, evaluated as $S_j = \lvert\gamma_j\rvert \cdot \lVert W_{:,j} \rVert_2$. |
| $\kappa$ | Fourth standardized moment kurtosis of channel saliency vector: $\kappa = \mathbb{E}[(S-\mu_S)^4]/\sigma_S^4$. |
| $r, \Omega$ | Protected outlier fraction $r \in [0.005, 0.03]$ and the corresponding decoupled channel index set $\Omega$. |
| $G = 32$ | Microscaled quantization group size along feature dimension $K$, matching GPU warp width. |
| $s, z, q$ | Local Group-32 scale factor $s \in \mathrm{FP16}$, packed zero-point $z \in [0, 15]$, and 4-bit integer code $q \in \{0, \dots, 15\}$. |
| $b(r)$ | Effective bitwidth per weight parameter: $b(r) = 4.625 + 16r$ bits (accounting for full-grid Group-32 packing of $4.625$ bpb plus $16r$ bpb for un-quantized FP16 outlier columns, evaluating to $4.785$ bpb at $r = 1\%$ and $4.945$ bpb at $r = 2\%$). |
| $L, H_{\text{kv}}, D_h, S, B$ | Transformer layers $L$, key-value heads $H_{\text{kv}}$, head dimension $D_h$, sequence length $S$, batch size $B$. |
| $t_c, t_x$ | Execution computation time ($t_c$) and PCIe direct memory access transfer time ($t_x$) per layer. |

Copyright (c) 2026 NeuraCryption Private Limited. All Rights Reserved. neuracryption.com/spectra-t&c

# STATUTORY INTELLECTUAL PROPERTY NOTICE AND GLOBAL LICENSING COVENANT

**PROPRIETARY ARCHITECTURAL SPECIFICATION, INDIVIDUAL NON-COMMERCIAL USE LICENCE, AND MANDATORY TERMS**
**GOVERNED EXCLUSIVELY UNDER THE LAWS OF INDIA AND INTERNATIONAL COPYRIGHT CONVENTIONS • neuracryption.com/spectra-t&c**

## STATUTORY OWNERSHIP AND MANDATORY TERMS NOTICE

The Sparse-Precision Exact Channel Tensor Reconstruction Architecture (SPECTRA), including the mathematical quantization formulations, Abstract Syntax Tree (AST) normalization saliency routing algorithms, Marlin byte-interleaved bit-packing layouts, TRDC (.trdc) binary container serialization formats, fused Triton GPU GEMV/GEMM microarchitectures, SpectraFusedEmbedding dynamic row-gather routines, and PingPongLayerOffloader execution pipelines disclosed in this research specification, constitute proprietary intellectual property owned solely and exclusively by NEURACRYPTION PRIVATE LIMITED (Corporate Identity Number: U62099PN2026PTC256069), having its registered office at Flat No. 401, Wing E, Spring Meadows No. 4/12/13/23, Ambegaon BK, Haveli, Pune 411046, Maharashtra, India (the "Licensor").

All access, inspection, download, reproduction, benchmarking, and utilization of this software, specification, runtime, and its compiled container weights are governed strictly by the NeuraCryption Global Terms of Use available at: **neuracryption.com/spectra-t&c**.

### 1. ACCEPTANCE OF TERMS

By using, downloading, copying, reading or accessing any part of the SPECTRA software, research specifications, documentation, container files or related materials (the "Materials"), you agree to be bound by this notice and the NeuraCryption Terms and Conditions of Website Use ("Website Terms") at neuracryption.com/spectra-t&c, which are incorporated into this notice. If this notice and the Website Terms conflict, the Website Terms prevail. Downloading, copying, installing, running, modifying or otherwise using any Materials is acceptance of the Website Terms. If you do not accept them, you must not use the Materials.

### 2. PERMITTED SCIENTIFIC AND NON-COMMERCIAL USE

The Licensor grants a personal, non-exclusive, non-transferable, non-sublicensable, revocable, royalty-free licence to natural persons, academic researchers, and peer-review evaluators, acting in an individual or academic capacity, to use the Materials for personal study, scientific reproduction, independent verification, and non-commercial academic research. The official open-source evaluation code and reference implementation are hosted at **https://github.com/NeuraCryption/SPECTRA**. No commercial licence is granted to any legal entity, corporation, or enterprise, and any commercial deployment or institutional distribution requires a separate written commercial agreement executed with the Licensor.

### 3. PROHIBITED ACTS

Except as Section 2 expressly permits, you must not: (a) make any commercial use, meaning any use that is directly or indirectly for revenue, profit, advantage or business operation, including use in a commercial product, service, API, hosted service, consulting, internal business process, or commercial model training; (b) monetise, sell, rent, lease, lend, sublicense, or redistribute the Materials for a fee; (c) reverse engineer, decompile, or disassemble proprietary source code or trade secrets beyond statutory interoperability exceptions; (d) create commercial clone implementations; (e) utilize the Materials to develop or distil competing commercial compression runtimes; (f) remove or alter any copyright mark; or (g) assist others in doing so. Scholarly citation, academic critique, and non-commercial peer-review benchmarking are expressly permitted.

### 4. OWNERSHIP AND RESERVATION OF RIGHTS

The Materials mean this report, the SPECTRA software in every version, its source and object code, GPU kernels, algorithms, methods (including HDSQ), the TRDC format and specification, documentation, benchmarks, and the names and marks SPECTRA, HDSQ, TRDC and NEURACRYPTION. They are the sole property of NEURACRYPTION PRIVATE LIMITED (CIN U62099PN2026PTC256069), registered office Flat No. 401, Wing E, Spring Meadows No. 4/12/13/23, Ambegaon BK, Haveli, Pune 411046, Maharashtra, India (the "Licensor"). Copyright © 2026 NeuraCryption Private Limited. All rights are reserved. Nothing grants, by implication, estoppel, exhaustion or otherwise, any licence to any patent, patent application, trade secret, trademark or other right. Publication of the Materials or of this report does not place them in the public domain and is not a waiver, dedication or disclosure to the public for any purpose beyond Section 2. Feedback or contributions you submit are licensed to the Licensor perpetually, irrevocably and worldwide without charge.

### 5. LAW THAT CANNOT BE EXCLUDED

The restrictions in this notice apply to the fullest extent permitted by applicable law. Where a law grants a right that cannot lawfully be excluded by contract (for example a limited right to observe or decompile for interoperability), it may be exercised only to the minimum extent and on the conditions that law prescribes, only after you have first requested the necessary information from the Licensor in writing, and any information obtained may be used for that purpose alone, must not be disclosed to others, and must not be used to develop a program that is substantially similar in expression or function. Any such exercise remains subject to Sections 3 and 8 in all other respects.

### 6. THIRD-PARTY MODELS AND MARKS

The Materials contain no third-party model weights and do not alter any third-party licence. Containers made from third-party weights remain subject to the licences of those weights, which you alone must obey. Third-party names appear only to identify tested models, imply no affiliation or endorsement, and belong to their owners.

### 7. TERMINATION

Your licence ends automatically and without notice on any breach, and the Licensor may revoke it at any time by notice. You must then stop all use and permanently delete all copies. Sections 3 to 6 and 8 to 13 survive termination.

### 8. STATUTORY REMEDIES

A breach of Sections 2 or 3 causes the Licensor harm that damages cannot adequately repair. The Licensor may seek interim and permanent injunctions without proof of special damage and, to the extent the law allows, without bond, in addition to damages, an account of profits and costs, and all civil and criminal remedies available under the Copyright Act, 1957 (including sections 51, 55, 63 and 63B), the Information Technology Act, 2000, the Indian Contract Act, 1872, and the equivalent laws of any country where the Materials are used. You will indemnify the Licensor against all losses, liabilities and costs, including legal fees, arising from your breach or from any claim by a person whom you allowed to use the Materials.

### 9. DISCLAIMER AND LIMIT OF LIABILITY

THE MATERIALS AND ALL RESULTS IN THIS REPORT ARE PROVIDED "AS IS" AND "AS AVAILABLE", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, ACCURACY, NON-INFRINGEMENT AND UNINTERRUPTED OR ERROR-FREE OPERATION. Measurements were taken on the stated test bed and may differ elsewhere. The Materials must not be used where failure could cause injury, loss of life or serious loss. To the maximum extent permitted by law, the Licensor is not liable for any direct, indirect, incidental, special, consequential or punitive loss, and its total liability for all claims shall not exceed INR 1,000. Nothing in this notice limits liability that cannot be limited by law.

### 10. GOVERNING LAW AND JURISDICTION

This notice and all disputes arising out of or in connection with the Materials are governed by the laws of India, without regard to conflict-of-law rules; the UN Convention on Contracts for the International Sale of Goods does not apply. The courts at Pune, Maharashtra, India have exclusive jurisdiction, and you submit to them irrevocably. The Licensor may also seek injunctive or protective relief in any court of competent jurisdiction anywhere in the world.

### 11. COMPLIANCE WITH LAWS

You must comply with all export-control, sanctions and data-protection laws that apply to your use and must not use the Materials for any unlawful purpose.

### 12. GENERAL PROVISIONS

If any provision is held invalid or unenforceable, it shall be enforced to the greatest extent permitted and limited or modified to the minimum extent needed, and the remaining provisions continue in full force. No failure or delay by the Licensor is a waiver. The Website Terms and this notice are the entire agreement on this subject. The Licensor may assign these terms freely; you may not. Headings do not affect meaning. The English text prevails over any translation.

### 13. LICENSING INQUIRIES AND CONTACT ROUTE

To request an entity, institutional or commercial licence, or to report misuse or infringement, submit inquiries via the official route at: **neuracryption.com/spectra-t&c**. Silence or the absence of enforcement by the Licensor shall never constitute consent, waiver, or estoppel.

CORPORATE HEADQUARTERS & REGISTERED OFFICE:
NEURACRYPTION PRIVATE LIMITED (CIN: U62099PN2026PTC256069)
Flat No. 401, Wing E, Spring Meadows No. 4/12/13/23, Ambegaon BK, Haveli, Pune 411046, Maharashtra, India.
Scientific Inquiries: contact@neuracryption.com • Commercial Licensing: legal@neuracryption.com

OFFICIAL TERMS REPOSITORY:
neuracryption.com/spectra-t&c
November 2026
Copyright © 2026 NeuraCryption Private Limited. All Rights Reserved Globally.