POTENTIAL–TIME DUAL REPRESENTATIONS FOR TTFS TRANSFORMER CONVERSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-to-first-spike (TTFS) coding represents each value by a single spike time, reducing the number of spikes required to represent an activation. Converting a pretrained Transformer into a TTFS-based continuous-time spiking neural network (SNN), however, requires composing Transformer operations from analog-compatible operators and evaluating the converted model under timing noise. We propose a Φ/Ψ conversion framework based on the dual relationship between membrane-potential space $\mathcal{V}$ and spike-time space $\mathcal{T}$. The framework composes a small set of fixed primitive operators with known hardware realizations to implement affine transformations, attention, LayerNorm/RMSNorm, GELU, and SwiGLU. Across pretrained ViT, RoBERTa, GPT-2, and Llama 2 7B models, conversion reduces classification accuracy by at most 0.11 pp (one fewer correct SST-2 prediction out of 872), while the absolute WikiText-2 perplexity change is below 0.00021. We model spike-time noise and the resulting deadline misses, parameterizing the noise with BrainScaleS-2 measurements from the physical paths implementing the linear and logarithmic ϕ operators. Code is available at `https://anonymous.4open.science/r/52AD`.

1 INTRODUCTION

ANN-to-SNN conversion offers a way to map pretrained Transformers to event-driven hardware without retraining. Among temporal coding schemes, TTFS is particularly attractive because each activation is represented by a single spike, reducing the number of spikes required to communicate an activation compared with rate-coded conversion (Mostafa, 2018; Göltz et al., 2021; Stanojevic et al., 2023). Extending continuous-time TTFS conversion from ReLU networks to Transformers, however, is nontrivial: Transformer blocks additionally require attention, normalization, and nonlinearities that are not directly provided by standard spiking dynamics (Zhao et al., 2025; Wang et al., 2026).

Existing TTFS Transformer conversions implement required nonlinear functions through specially engineered current kernels or enriched neuron dynamics (Zhao et al., 2025; Wang et al., 2026). This leaves open a more basic question: can Transformer computation instead be synthesized by composing a fixed set of operators across two constrained representations, membrane potential and spike time?

Our key observation is that continuous-time TTFS computation already involves these two representations: a computational value is stored as membrane potential, exposed as spike timing, and interpreted again as membrane potential at the receiving neuron. Rather than designing new dynamics for each target function, we decompose the mappings between membrane-potential and spike-time spaces into a small set of composable operators. We formalize this potential–time dual representation as a Φ/Ψ framework and show that a small set of primitive operators is sufficient to synthesize the Transformer operations considered here.

The ideal operator compositions, however, assume unrestricted membrane-potential and spike-time ranges, whereas physical implementations operate over finite potential ranges and observation windows. We therefore calibrate layer-wise potential ranges and bound each spike-time window to accommodate finite representational ranges. Across the evaluated vision and language models, the converted networks closely preserve their ANN counterparts, with at most 0.115 percentage-point degradation in classification accuracy and negligible change in language-model perplexity.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

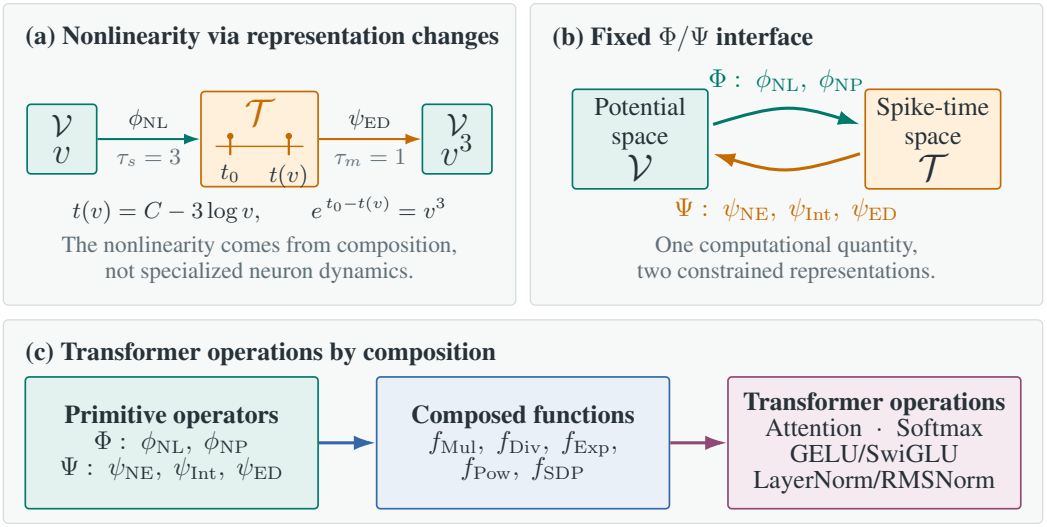


Figure 1: Overview of the Φ/Ψ framework. (a) Encoding a potential v with ϕ_{NL} ($\tau_s = 3$) and decoding the spike-time offset with ψ_{ED} ($\tau_m = 1$) yields v^3. (b) Φ maps membrane potentials to spike times, and Ψ maps them back. (c) Primitive operators compose into the functions of Table 2 and then into Transformer operations.

Prior TTFS work has examined finite temporal resolution and timing perturbations (Stanojevic et al., 2024). We additionally consider deadline misses, where a perturbed event falls outside the observation window. We evaluate the sensitivity of converted models to timing perturbations and deadline misses, and reduce misses using an observation margin inspired by prior TTFS conversion (Stanojevic et al., 2023). We use BrainScaleS-2 measurements to set timing-noise levels in this evaluation.

In summary, this study (1) abstracts the potential–time dual representation into a Φ/Ψ framework and constructs Transformer operations by composing a small set of primitive operators; (2) bounds spike-time windows through potential calibration; and (3) evaluates model sensitivity to timing noise parameterized by hardware measurements.

Figure 1 summarizes the central idea: a computational quantity alternates between membrane-potential and spike-time representations, and nonlinear operations emerge by composing fixed operators across the two spaces.

2 Related Work

ANN-to-SNN conversion. A converted SNN reuses trained ANN parameters, avoiding retraining while targeting event-driven inference (Rueckauer et al., 2017; Sengupta et al., 2019; Deng & Gu, 2021). Rate-coded conversion (Rueckauer et al., 2017; Deng & Gu, 2021) replaces ReLU with firing-rate proxies and scales thresholds to match activation statistics, but uses many spikes to represent each value. Temporally coded conversion (Stanojevic et al., 2023; Zhao et al., 2025) instead represents each activation by the timing of a single spike. Stanojevic et al. (2023) proved an exact conversion for ReLU networks but did not extend it to Transformer nonlinearities. TTFSFormer (Zhao et al., 2025) implements log, exp, and GELU functions with specially engineered current kernels. We complement this line of work by defining a fixed set of operators over potential and spike-time representations, reporting synaptic operation counts under an explicit mapping, and evaluating model sensitivity to Gaussian timing noise parameterized by measurements of physical paths implementing the ϕ operators. Wang et al. (2026) used multi-basis exponential decay neurons with distinct decay schedules to approximate functions and convert Transformers. Yuan et al. (2026) combine LIF-based division, shift-based norm approximation, and lookup-based exponential approximation to construct Transformer nonlinearities. Our framework composes a fixed set of primitive operator types between membrane-potential and spike-time representations in continuous time.

Transformer architectures with spiking neurons. A growing line of work builds Transformers directly in the spiking domain. Spikformer (Zhou et al., 2023) replaces attention with spiking self-attention based on LIF neurons. Spike-driven Transformer (Yao et al., 2023) removes multiplications through logical-AND-style operations. Both operate in discrete time and use rate- or direct-coded activations for digital neuromorphic hardware. SpikeGPT (Zhu et al., 2023) extends spiking Transformers to language modeling with discrete-time spike-driven blocks. We instead study continuous-time temporal coding on analog neuromorphic substrates, where computation depends directly on spike timing and continuous neuron dynamics.

Analog neuromorphic hardware. Analog neuromorphic chips (Benjamin et al., 2014; Rubino et al., 2023; Richter et al., 2024) implement neuron dynamics directly in subthreshold CMOS, allowing RC circuits to compute the governing differential equations without fetch-decode-execute cycles. Recent analog platforms support LIF neurons and short-term plasticity (Richter et al., 2024), as well as multi-compartment models with dendritic plateau potentials (Pehle et al., 2022). Efficiently realizing operations such as Softmax, LayerNorm, and GELU remains an operator-design challenge on analog neuromorphic substrates. Our Φ/Ψ framework addresses this operator-composition aspect by identifying a small analog-compatible interface for Transformer computation.

3 Preliminaries

Transformer ingredients. For input $\mathbf{z} \in \mathbb{R}^{N\times D}$, let $\mathcal{A}(\mathbf{z}) = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_h)\mathbf{W}^O$ denote multi-head self-attention, where $\mathrm{head}_i = \mathrm{Softmax}(\mathbf{Q}_i\mathbf{K}_i^\top/\sqrt{d_k})\mathbf{V}_i$ with head dimension d_k and optional causal masking, and let $\mathcal{F}(\mathbf{z}) = \sigma(\mathbf{z}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2$ denote an ungated feed-forward MLP with $\sigma \in \{\mathrm{ReLU}, \mathrm{GELU}\}$. Pre-norm and post-norm blocks with LayerNorm (LN) are respectively

$$\begin{aligned} \text{pre-norm:} \quad & \mathbf{u} = \mathbf{z} + \mathcal{A}(\mathrm{LN}(\mathbf{z})), \quad \mathbf{z}' = \mathbf{u} + \mathcal{F}(\mathrm{LN}(\mathbf{u})), \\ \text{post-norm:} \quad & \mathbf{u} = \mathrm{LN}(\mathbf{z} + \mathcal{A}(\mathbf{z})), \quad \mathbf{z}' = \mathrm{LN}(\mathbf{u} + \mathcal{F}(\mathbf{u})). \end{aligned} \tag{1}$$

ViT and GPT-2 use pre-norm, whereas BERT and RoBERTa use post-norm. Llama 2 uses pre-norm with RMSNorm and a SwiGLU MLP (Appendices B.4 and B.5). A full Transformer stacks L such blocks around an input embedding and an output head. Both orderings therefore share attention, normalization, activation, linear, and residual operations.

Spiking neuron models. We consider the leaky integrate-and-fire (LIF) model and its integrate-and-fire (IF) limit as $\tau_m \to \infty$. With membrane time constant τ_m and synaptic weights w_i, the dynamics are

$$\underbrace{\tfrac{dV}{dt} = -\tfrac{V}{\tau_m} + \sum_i w_i I_i(t),}_{\text{LIF}} \qquad \underbrace{\tfrac{dV}{dt} = \sum_i w_i I_i(t)\,.}_{\text{IF}} \tag{2}$$

When $V(t)$ reaches the threshold θ, the neuron emits a spike and resets. We use the following synaptic current kernel, where τ_s is the synaptic time constant:

$$I(t) = \exp\left(-\frac{t}{\tau_s}\right)\Theta(t), \tag{3}$$

with Θ the Heaviside step function. For a spike at time t_{in} and $\tau_m \neq \tau_s$, the spike-evoked component of the LIF membrane potential at $t \geq t_{\text{in}}$ has the form

$$V(t) \propto \exp\left(-\frac{t - t_{\text{in}}}{\tau_m}\right) - \exp\left(-\frac{t - t_{\text{in}}}{\tau_s}\right). \tag{4}$$

This is the standard double-exponential postsynaptic potential (PSP). It approaches a single exponential with time constant τ_m when $\tau_s \ll \tau_m$ and $t - t_{\text{in}} \gg \tau_s$, or with time constant τ_s when $\tau_m \ll \tau_s$ and $t - t_{\text{in}} \gg \tau_m$.

4 Potential–Time Duality and the Φ/Ψ Framework

Consider an IF neuron whose initial membrane potential stores a value $0 < v \leq 1$, with firing threshold $\theta = 1$. Starting at reference time C, we apply the exponentially decaying current kernel

in Equation (3) with $\tau_s = 3$, using a synaptic weight w such that $w\tau_s = \theta$. The threshold-crossing condition then gives

$$t = C - 3\log v. \tag{5}$$

Applying the same mapping to the reference value 1 produces a spike at $t_0 = C$. Now let these two spike times drive a receiving neuron whose membrane response depends exponentially on their temporal difference, with membrane time constant $\tau_m = 1$. Its resulting contribution is

$$\exp(t_0 - t) = \exp(C - (C - 3\log v)) = v^3. \tag{6}$$

The cubic nonlinearity therefore emerges from composing mappings between two representations rather than from specialized neuron dynamics. Spike timing both conveys a membrane value and participates in computation. The result depends jointly on how a membrane potential is exposed as a spike time and how that spike time is interpreted back as a membrane potential. We formalize this observation by distinguishing two classes of mappings. Let Φ denote mappings from membrane-potential space $\mathcal{V}$ to spike-time space $\mathcal{T}$, and let Ψ denote mappings from $\mathcal{T}$ back to $\mathcal{V}$:

$$\mathcal{V} \xrightarrow{\phi \in \Phi} \mathcal{T} \xrightarrow{\psi \in \Psi} \mathcal{V}. \tag{7}$$

We refer to this pair of representations as the *potential–time dual representation*. The cubic mapping is one particular composition of these two types of mappings. More generally, changing the dynamics used on either side of the temporal representation yields different mappings. For example, if the time constants of the ϕ and ψ operations are τ_s and τ_m, respectively, the same construction produces

$$v^{\tau_s/\tau_m}. \tag{8}$$

This suggests treating the underlying neuron and synaptic dynamics not as isolated mechanisms for individual functions, but as a small compositional interface between potential and temporal representations.

The arithmetic functions listed below and the Transformer operations evaluated in this work are realized by alternating between the two representations through compositions of Φ and Ψ operators.

Primitive operators. Table 1 lists the fixed primitive interface used in this work. Each primitive operator abstracts a computation that has been demonstrated in prior analog hardware. For example, ϕ_{NP} is implemented by integrating a constant current in BrainScaleS-2 and Dynap-SE2 (Pehle et al., 2022; Richter et al., 2024), while ψ_{Int} is implemented through pulse-width integration (Nägele et al., 2023; 2024). Implementations of ψ_{ED} have also been demonstrated in prior analog hardware (Greatorex et al., 2025; Pehle et al., 2022).

Table 1: Primitive Φ/Ψ operators. NL (Negative-Log), NP (Negative-Potential), NE (Negative-Exponential), Int (Integration), and ED (Exponential-Difference) denote the corresponding operators, respectively.

Operator	Type	Mapping	Implementation
ϕ_{NL}	Φ	$v \mapsto C - \tau_s \log v$	IF neuron + exponential response
ϕ_{NP}	Φ	$v \mapsto \theta - v$	Pehle et al. (2022); Richter et al. (2024)
ψ_{NE}	Ψ	$t_{\mathrm{in}} \mapsto \exp(-(t - t_{\mathrm{in}})/\tau_m)$	LIF neuron + Dirac delta response
ψ_{Int}	Ψ	$t_1, t_2; v \mapsto \int_{t_1}^{t_2} I(v)ds$	Nägele et al. (2023; 2024)
ψ_{ED}	Ψ	$t_1, t_2; \tau_m \mapsto \exp((t_2 - t_1)/\tau_m)$	Greatorex et al. (2025); Pehle et al. (2022)

Compositions of primitive operators. Table 2 lists functions composed from the primitive operators. Because each Ψ operator has a synaptic weight that scales its output, fixed coefficients in these compositions are absorbed into the receiving synaptic gains. In the GELU composition, f_{Pow} uses the two signed branches described in Section 5.2, while the final product of input and gate remains f_{Mul}. These compositions also serve as the building blocks for Softmax and LayerNorm. The negative sign in f_{SDP} compensates for $f_{\mathrm{Exp}}(v) = \exp(-v)$, yielding the positive scaled dot products used by Softmax after the two operators are composed. The noise-free identities and finite-domain conditions for the composed operators are derived in Appendix B. Together, these constructions show that the Transformer operations considered here can be synthesized within the same fixed primitive interface, without introducing function-specific neuron dynamics or current kernels.

Table 2: Compositions of the primitive operators. $\mathrm{GELU}(v) \approx \frac{1}{2}v[1+\tanh(\sqrt{2/\pi}(v+cv^3))]$ is the cubic tanh approximation used in the primitive operator compositions, with $c = 0.044715$; the logistic form shown in the table is algebraically equivalent. To distinguish fixed synaptic gains from composed multiplication, we write $\alpha \cdot_\Psi y$ for a contribution y delivered with gain α by the receiving Ψ operator. This notation does not denote a separate invocation of f_{Mul} or an additional spike event.

Operator	Mapping	Components
f_{Mul}	$v_1, v_2 \mapsto v_1 v_2$	$\psi_{\mathrm{Int}}(\phi_{\mathrm{NP}}(v_2), \phi_{\mathrm{NP}}(0); v_1)$
f_{Div}	$v_1, v_2 \mapsto v_1/v_2$	$\psi_{\mathrm{ED}}(\phi_{\mathrm{NL}}(v_1;\tau_s), \phi_{\mathrm{NL}}(v_2;\tau_s);\tau_s)$
f_{Exp}	$v \mapsto \exp(-v)$	$\psi_{\mathrm{NE}} \circ \phi_{\mathrm{NP}}(v)$
f_{Pow}	$v;\tau_s,\tau_m \mapsto v^{\tau_s/\tau_m}$	$\psi_{\mathrm{ED}}(\phi_{\mathrm{NL}}(v;\tau_s), \phi_{\mathrm{NL}}(1;\tau_s);\tau_m)$
f_{GELU}	$v \mapsto v/(1+e^{-z})$, where $z = \sqrt{8/\pi}\,(v+cv^3)$	$f_{\mathrm{Mul}}\big(v,\ f_{\mathrm{Div}}(1,\ 1+f_{\mathrm{Exp}}(z))\big)$, where $z = \sqrt{8/\pi} \cdot_\Psi v + c\sqrt{8/\pi} \cdot_\Psi f_{\mathrm{Pow}}(v; 3\tau_m, \tau_m)$
f_{SDP}	$\mathbf{q},\mathbf{k} \mapsto -\langle \mathbf{q},\mathbf{k}\rangle/\sqrt{d_k}$	$-d_k^{-1/2} \cdot_\Psi \sum_{r=1}^{d_k} f_{\mathrm{Mul}}(q_r, k_r)$

5 METHODOLOGY

5.1 CALIBRATION AND FINITE POTENTIAL RANGES

The operator equations in Section 4 hold exactly within their stated and unbounded domains in an ideal continuous-time system. The choice of domain sets the clipping boundaries and spike-time windows and therefore affects the accuracy of the finite-domain approximation. Finite implementations cannot represent unbounded membrane potentials or spike times. For example, applying ϕ_{NP} with threshold $\theta = 1$ to a membrane potential of -0.5 gives $t = 1 - (-0.5) = 1.5$. As the membrane potential decreases, the output spike time grows without bound. Residual additions can widen activation ranges across Transformer layers, so this issue also arises in converted models.

We therefore bound the input potential range of each ϕ operator, which also bounds its spike-time window. The same requirement applies to ϕ_{NL}: for an input of e^{-6} and $\tau_s = 3$, the firing time is $t = C - 3\log e^{-6} = C + 18$.

Finite potential and spike-time ranges clip inputs outside their declared bounds. Information loss due to small values in a finite time window has also been noted (Park et al., 2020). We therefore assign fixed potential ranges and evaluate the resulting conversion fidelity.

Three sources of potential ranges. There are three types of potential ranges we need to handle. The first is when a function has a fixed range, such as softmax. For example, softmax outputs always exist within the range $[0, 1]$; in this case, we simply use this fixed range as is. Second, there are cases where the range is computable but exceeds the input range, such as with linear or attention layers. In this case, we constrain the range through calibration at the necessary layers to prevent the range from expanding. Finally, there are cases where the output range is not finite, such as with ϕ_{NL}. In this case, we first restrict the representable input range. Specifically, if the positive input range is $[V_{\min}, V_{\max}]$, where $V_{\min} > 0$, the time window length is set to $T = \tau_s \log(V_{\max}/V_{\min})$. Cases where the interval is computable and the output range is smaller than the input range (such as GELU) are excluded from this discussion, as they can be handled by combining the input range with a fixed range. We apply calibration only to the second of these three cases.

Layer-wise calibration. Calibration passes a fixed subset of the training data through the converted model and records the value range and distribution at each calibrated location. The range endpoints are selected and expanded from these statistics (Appendix A) and remain fixed during inference; values outside them are clamped. Analytic bounds and the positive lower bound for ϕ_{NL} are specified separately.

5.2 POWER OPERATOR

The power composition in Table 2 is defined for positive inputs. For positive integer powers of a signed input v, we use two branches carrying v and $-v$, with the sign reversal implemented by the

receiving synaptic gain. Excluding contributions from non-firing branches, the outputs combine as

$$v^p = \big(\max(v, 0)\big)^p + (-1)^p \big(\max(-v, 0)\big)^p.$$

Thus, for even powers, the two branch outputs add with the same sign, whereas for odd powers, the negative branch combines with a negative receiving gain.

5.3 Primitive-Level Error Model

Timing variation affects the proposed computation in two related ways. First, shifting a spike time changes the value recovered by the receiving Ψ operator. Second, a sufficiently late spike can cross the observation deadline and become unavailable, producing an error that depends on the receiving operator rather than an additional timing offset. We model both effects and evaluate how they propagate through the converted model.

Spike-time noise and deadline misses. For each spike time t, we sample $\epsilon \sim \mathcal{N}(0, \sigma_t^2)$ and set $\tilde{t} = t + \epsilon$. A spike is omitted if $\tilde{t}$ exceeds the observation deadline. A deadline miss prevents the receiving operator from observing the input spike, but the resulting change depends on the operator, as the following two cases illustrate.

For ψ_{Int}, let the observation deadline be $T = 1$ and let the operator receive $t_1 = 0.5$ and $t_2 = T$. If noise moves t_2 beyond the deadline, integration remains limited to the observation window, and the output remains $\int_{0.5}^{1} I(s)\, ds$.

In contrast, let ψ_{NE} receive $t_{\mathrm{in}} = 1$ with observation time $t = 1$ and $\tau_m = 1$. If the membrane potential responds immediately at spike arrival, the output is $w \exp(-(1-1)/1) = w$. Positive timing noise changes the arrival time to $1 + \epsilon$, beyond the observation time, so the output becomes zero.

BrainScaleS-2 timing scales. On BrainScaleS-2, we measure timing variation across repeated presentations of each input code along two physical paths, each combining synaptic loading with either the ϕ_{NP} or ϕ_{NL} operation (Pehle et al., 2022). The NP timing scale is exploratory; Appendix E details the circuit selection and measurement limitations. For path $e \in \{\mathrm{NP}, \mathrm{NL}\}$, let $r_{t,e}$ be the measured timing standard deviation normalized by that path's fitted signal span. At application ℓ of a ϕ operator in the model, we assume this standard deviation scales with the declared local time-window width $T_{e,\ell}$ and set

$$\sigma_{t,e,\ell} = \alpha r_{t,e} T_{e,\ell}, \qquad \epsilon_{e,\ell} \sim \mathcal{N}(0, \sigma_{t,e,\ell}^2). \tag{9}$$

The common factor α scales both timing-noise fractions while preserving their ratio. We sample independent Gaussian perturbations at the corresponding ϕ operator applications and leave the other Ψ operations ideal. Appendix E details the measurement and injection protocol.

Observation margin. To mitigate deadline misses, we introduce a fixed observation margin between the maximum spike time and the observation time, following the safety-margin idea used in prior work (Stanojevic et al., 2023). Under simulated Gaussian timing noise, this margin sharply reduces deadline misses (Appendix D.3).

6 Experiment

We evaluate conversion fidelity and operation cost, compatibility with discrete-time simulation, and sensitivity to timing noise parameterized by BrainScaleS-2 measurements.

6.1 Experimental Setup

Pretrained models and datasets. We implement the converted models in PyTorch and Hugging Face Transformers (Paszke et al., 2019; Wolf et al., 2019) and evaluate their operators using continuous spike times. We evaluate ViT (Dosovitskiy et al., 2020) on CIFAR-10 and ImageNet-1k (Deng et al., 2009), RoBERTa on SST-2 (Socher et al., 2013), and GPT-2 (Radford et al., 2019) and Llama 2 7B (Touvron et al., 2023) on WikiText-2 (Merity et al., 2017). For the timing-noise

comparison, we additionally evaluate CCT-7/3×1 (Hassani et al., 2021) on CIFAR-10. All models use publicly available pretrained checkpoints (Appendix A), with the ImageNet-1k ViTs taken from timm (Wightman, 2019).

Baselines. We compare each converted model with its source full-precision ANN and with two groups of prior conversion methods: discrete-time, direct-coded SNNs designed for digital hardware, namely SpikeZIP-TF (You et al., 2024) and the multi-basis exponential decay neurons of Wang et al. (2026); and continuous-time TTFS SNNs, namely TTFSFormer (Zhao et al., 2025) and the exact ReLU mapping of Stanojevic et al. (2023).

Conversion hyperparameters. For continuous-time evaluation, the membrane time constant is set to $\tau_m = 1$. The representable potential ranges are fixed using analytically defined bounds or layer-wise calibration. We set the base synaptic time constant to $\tau_s = 1$; power and normalization compositions rescale it for their ϕ_{NL} operations (Section 5.2 and Appendix B.4). Calibration uses a fixed subset of 5,000 training examples and no labels or evaluation data (Appendix A).

6.2 Conversion Fidelity and Operation Cost

We first evaluate the conversion before introducing primitive-level errors. Because prior studies use different checkpoints and evaluation settings, absolute accuracy is not directly comparable across rows. We therefore use the matched ANN-to-SNN difference, $\Delta = \mathrm{SNN} - \mathrm{ANN}$, as a within-row measure of conversion fidelity.

Table 3: Comparison with prior ANN-to-SNN conversions on image classification. ANN and SNN report top-1 accuracy (%), and Ops. reports 10^9 synaptic operations. T denotes the number of time steps; Cont. denotes continuous time. †SpikeZIP-TF applies 32-level quantization-aware training before conversion (SViT-S/B/L-32L in You et al. (2024)); we list these results under their underlying architectures.

Dataset	Arch.	Method	T	Coding	ANN	SNN	Δ	Ops.
CIFAR-10	VGG16	Stanojevic et al. (2023)	Cont.	TTFS	93.59	93.59	0.00	–
	ViT-S/16	SpikeZIP-TF† (You et al., 2024)	32	Direct	99.2	98.7	−0.50	–
	ViT-S/16	**Ours**	Cont.	TTFS	98.48	98.47	−0.01	4.75
ImageNet-1k	ViT-S/16	SpikeZIP-TF† (You et al., 2024)	64	Direct	82.34	81.45	−0.89	–
		TTFSFormer (Zhao et al., 2025)	Cont.	TTFS	81.38	81.40	+0.02	5.42
		Ours	Cont.	TTFS	82.28	82.28	0.00	4.75
	ViT-B/16	SpikeZIP-TF† (You et al., 2024)	64	Direct	83.75	82.71	−1.04	–
		Wang et al. (2026)	16	Direct	83.44	83.00	−0.44	–
		TTFSFormer (Zhao et al., 2025)	Cont.	TTFS	85.10	85.07	−0.03	19.20
		Ours	Cont.	TTFS	86.06	86.00	−0.06	17.87
	ViT-L/16	SpikeZIP-TF† (You et al., 2024)	64	Direct	85.41	83.82	−1.59	–
		TTFSFormer (Zhao et al., 2025)	Cont.	TTFS	85.83	85.78	−0.05	65.70
		Ours	Cont.	TTFS	86.38	86.38	0.00	62.36

Image classification. Table 3 compares ANN-to-SNN conversion methods on CIFAR-10 and ImageNet-1k. Our CIFAR-10 result uses the 10k-image test set, and our ImageNet-1k results use a fixed 5k-image validation subset. Results from prior studies retain the evaluation settings reported in their respective papers. Across the evaluated ViTs, conversion reduces top-1 accuracy by at most 0.06 pp for ImageNet-1k ViT-B. Thus, the converted models preserve the performance of their source ANNs while using the proposed operator compositions.

Natural language processing. Table 4 reports conversion fidelity on SST-2 and WikiText-2. For RoBERTa-B, ANN predicts 824/872 and SNN 823/872 SST-2 examples correctly. RoBERTa-L accuracy is unchanged to two decimal places. On WikiText-2, GPT-2 perplexity rises by 0.000207, whereas Llama 2 7B perplexity changes from 12.015068 to 12.015066 (-2.27×10^{-6}). The larger

Table 4: Conversion fidelity on natural-language tasks: SST-2 accuracy (higher is better) and WikiText-2 perplexity (lower is better). Δ is measured in percentage points for SST-2 and perplexity units for WikiText-2. For our rows, Δ is computed from unrounded results before display rounding.

Task	Architecture	Method	T	Coding	ANN	SNN	Δ
SST-2	RoBERTa-B	SpikeZIP-TF (You et al., 2024)	64	Direct	94.15%	92.81%	−1.34
		Wang et al. (2026)	16	Direct	94.49%	93.46%	−1.03
		Ours	Cont.	TTFS	94.50%	94.38%	−0.11
	RoBERTa-L	SpikeZIP-TF (You et al., 2024)	128	Direct	96.21%	93.79%	−2.42
		Wang et al. (2026)	16	Direct	96.22%	95.98%	−0.24
		Ours	Cont.	TTFS	96.44%	96.44%	0.00
WikiText-2	GPT-2	Wang et al. (2026)	16	Direct	22.34	22.69	+0.35
		Ours	Cont.	TTFS	21.98	21.98	0.00
	Llama 2 7B	**Ours**	Cont.	TTFS	12.02	12.02	0.00

absolute perplexity change is below 0.00021. Appendix D.4 reports a separate Llama timing-noise diagnostic.

Operation cost. Ops. in Table 3 counts, per inference, every spike delivered to a destination together with the synchronization and reference events of each ϕ operation (Appendix C). Under this count, our ViT-S, ViT-B, and ViT-L conversions require 12%, 7%, and 5% fewer operations than TTFSFormer.

6.3 Discrete-time Simulation Compatibility

The proposed conversion uses continuous spike times and targets analog neuromorphic hardware. Its LIF/IF dynamics, time-dependent current integration, and exponential decay can also be evaluated in discrete-time simulation. Stanojevic et al. (2024) show that additional training with spike times restricted to discrete time steps substantially recovers TTFS accuracy at low temporal resolution. Our sweep instead discretizes the converted pretrained model without additional training or recalibration, isolating the accuracy loss caused by finite temporal resolution.

Figure 2 compares a fixed time interval with a fixed number of steps per ϕ operator window, using the same model and calibration, double precision, and no timing noise. At 2,048 steps per window, accuracy matches the $\Delta t = 0.01$ run (86.2% for both). Appendix D.1 reports the distribution of steps per window under a fixed time interval.

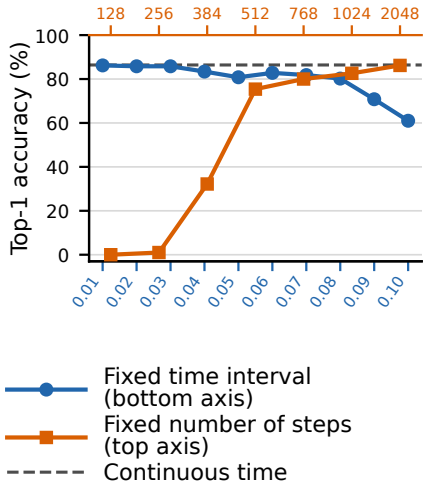
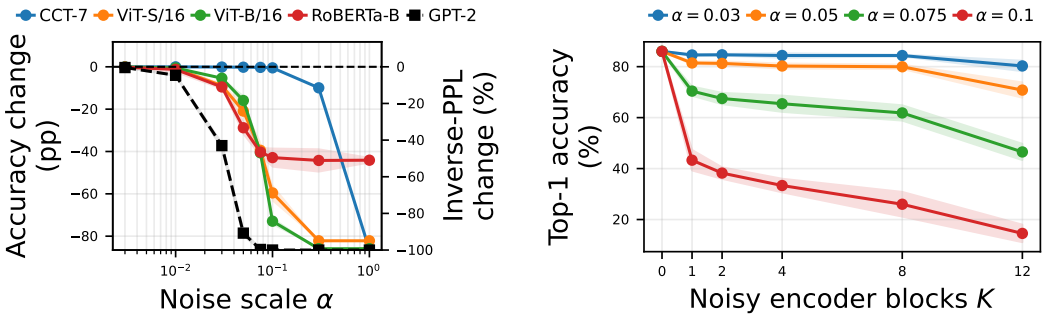


Figure 2: Discrete-time ViT-B/16 accuracy on 500 ImageNet-1k validation images.

6.4 Sensitivity to Timing Noise Parameterized by BrainScaleS-2 Measurements

We use the scaling in Equation (9) as a modeling assumption linking measured timing variation to local model time windows. Figure 3(a) compares model sensitivity under this assumption. At the measured scale ($\alpha = 1$), all four classifiers approach chance accuracy and GPT-2 inverse perplexity decreases by nearly 100%, whereas ViT-B/16 retains 85.45% accuracy at $\alpha = 0.01$. These results characterize sensitivity under the scaling assumption, rather than predict accuracy on BrainScaleS-2; Appendix D.2 gives the measurement values and evaluation settings.



(a) Timing-noise sensitivity across models

(b) Noise in the first K encoder blocks

Figure 3: Timing-noise sensitivity under Equation (9). (a) Classifiers report accuracy changes from their deterministic converted baselines on the left axis; GPT-2 reports the relative change in inverse perplexity, $100(\mathrm{PPL}_{\mathrm{clean}}/\mathrm{PPL}_{\mathrm{noisy}} - 1)$, on the right. (b) ViT-B/16 top-1 accuracy with noise in the first K encoder blocks. Curves show means and 95% Student-t intervals over three noise seeds, with an observation margin of $4\sigma_t$.

Figure 3(b) varies the number of initial ViT-B/16 encoder blocks receiving noise at both ϕ operators. Embeddings, final normalization, and the classifier receive no injected noise; $K = 0$ supplies the deterministic reference.

At $\alpha = 0.075$, mean top-1 accuracy falls from 70.41% with one perturbed block to 46.53% with all twelve. Increasing the number of perturbed blocks compounds the accuracy loss. This experiment measures the cumulative effect of perturbing progressively more blocks, rather than the isolated effect of individual blocks. In a separate diagnostic, disabling timing noise in the MLP linear projections gives the largest accuracy recovery (Appendix D.2).

7 Conclusion

We propose a Φ/Ψ framework for converting pretrained ANN Transformers into SNNs. The framework alternates between membrane-potential and spike-time representations and composes fixed primitive operators to implement Transformer operations. We also evaluate model sensitivity to timing noise parameterized by BrainScaleS-2 measurements. By restricting the construction to primitives with known analog realizations, we convert the evaluated GELU- and SwiGLU-based Transformers with the same small fixed operator set and obtain operation counts comparable to prior conversion methods.

Our SOP model excludes hardware- and system-level implementation costs, and our robustness results are simulations based on limited hardware measurements. Future work should evaluate complete model execution on hardware and training methods that improve robustness to hardware non-idealities.

AI use statement

In this work, we used generative AI tools to assist with writing and checking the derivations of the operator identities, designing experiments, implementing methods, translating and editing manuscript text, and interpreting experimental results. We also used these tools to search for relevant literature and to write Python scripts that generate figures and tables. We worked through the operator identities and wrote the initial manuscript drafts ourselves. An author checked the AI-assisted derivations, reviewed the equations, code, results, figures, tables, and manuscript text, and verified literature identified with AI against the original sources. We take responsibility for the final content of this work, including text, claims, code, figures, and tables produced with the aid of generative AI.

Ethics statement

This work converts publicly available pretrained models and evaluates them on public datasets. It involves no human subjects or personal data, and we do not foresee ethical concerns beyond those of the underlying models.

Reproducibility statement

Code is available at `https://anonymous.4open.science/r/52AD`. Appendix A lists checkpoints, data populations, and calibration settings; Appendix B derives every operator composition; Appendix C gives the operation-count model; and Appendix E describes the hardware measurement protocol.

References

Ben Varkey Benjamin, Peiran Gao, Emmett McQuinn, Swadesh Choudhary, Anand R Chandrasekaran, Jean-Marie Bussat, Rodrigo Alvarez-Icaza, John V Arthur, Paul A Merolla, and Kwabena Boahen. Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proceedings of the IEEE*, 102(5):699–716, 2014.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 248–255. IEEE Computer Society, 2009.

Shikuang Deng and Shi Gu. Optimal conversion of conventional artificial neural networks to spiking neural networks. In *International Conference on Learning Representations (ICLR)*, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Julian Göltz, Laura Kriener, Andreas Baumbach, Sebastian Billaudelle, Oliver Breitwieser, Benjamin Cramer, Dominik Dold, Akos Ferenc Kungl, Walter Senn, Johannes Schemmel, et al. Fast and energy-efficient neuromorphic deep learning with first-spike times. *Nature Machine Intelligence*, 3 (9):823–835, 2021.

Hugh Greatorex, Michele Mastella, Ole Richter, Madison Cotteret, Willian Soares Girão, Ella Janotte, and Elisabetta Chicca. A scalable event-driven spatiotemporal feature extraction circuit. *arXiv preprint arXiv:2501.10155*, 2025.

Ali Hassani, Steven Walton, Nikhil Shah, Abulikemu Abuduweili, Jiachen Li, and Humphrey Shi. Escaping the big data paradigm with compact transformers. *arXiv preprint arXiv:2104.05704*, 2021.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Hesham Mostafa. Supervised learning based on temporal coding in spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 29(7):3227–3235, 2018.

Raphael Nägele, Jakob Finkbeiner, Valentin Stadtlander, Markus Grözing, and Manfred Berroth. Analog multiply-accumulate cell with multi-bit resolution for all-analog AI inference accelerators. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 70(9):3509–3521, 2023.

Raphael Nägele, Jakob Finkbeiner, Markus Grözing, Manfred Berroth, and Georg Rademacher. Characterization of an analog MAC cell with multi-bit resolution for AI inference accelerators. In *2024 IEEE 6th International Conference on AI Circuits and Systems (AICAS)*, pp. 243–247. IEEE, 2024.

Seongsik Park, Seijoon Kim, Byunggook Na, and Sungroh Yoon. T2FSNN: Deep spiking neural networks with time-to-first-spike coding. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6. IEEE, 2020.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

Christian Pehle, Sebastian Billaudelle, Benjamin Cramer, Jakob Kaiser, Korbinian Schreiber, Yannik Stradmann, Johannes Weis, Aron Leibfried, Eric Müller, and Johannes Schemmel. The BrainScaleS-2 accelerated neuromorphic system with hybrid plasticity. *Frontiers in Neuroscience*, 16:795876, 2022.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019. URL https://cdn.openai.com/better-language-models/language-models.pdf.

Ole Richter, Chenxi Wu, Adrian M Whatley, German Köstinger, Carsten Nielsen, et al. DYNAP-SE2: a scalable multi-core dynamic neuromorphic asynchronous spiking neural network processor. *Neuromorphic Computing and Engineering*, 4(1):014003, 2024.

Arianna Rubino, Matteo Cartiglia, Melika Payvand, and Giacomo Indiveri. Neuromorphic analog circuits for robust on-chip always-on learning in spiking neural networks. In *2023 IEEE 5th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pp. 1–5. IEEE, 2023. doi: 10.1109/AICAS57966.2023.10168620.

Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682, 2017.

Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. Going deeper in spiking neural networks: VGG and residual architectures. *Frontiers in Neuroscience*, 13:95, 2019.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1170.

Ana Stanojevic, Stanisław Woźniak, Guillaume Bellec, Giovanni Cherubini, Angeliki Pantazi, and Wulfram Gerstner. An exact mapping from ReLU networks to spiking neural networks. *Neural Networks*, 168:74–88, 2023.

Ana Stanojevic, Stanisław Woźniak, Guillaume Bellec, Giovanni Cherubini, Angeliki Pantazi, and Wulfram Gerstner. High-performance deep spiking neural networks with 0.3 spikes per neuron. *Nature Communications*, 15(1):6793, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Jingya Wang, Xin Deng, Wenjie Wei, Dehao Zhang, Shuai Wang, Qian Sun, Jieyuan Zhang, Hanwen Liu, Ning Xie, and Malu Zhang. Training-free ann-to-snn conversion for high-performance spiking transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 2128–2136, 2026.

Ross Wightman. PyTorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. HuggingFace's Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Man Yao, JiaKui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 64043–64058. Curran Associates, Inc., 2023. doi: 10.52202/075280-2798. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/ca0f5358dbadda74b3049711887e9ead-Paper-Conference.pdf.

Kang You, Zekai Xu, Chen Nie, Zhijie Deng, Qinghai Guo, Xiang Wang, and Zhezhi He. SpikeZIP-TF: Conversion is all you need for transformer-based SNN. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57367–57383. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/you24b.html.

Xinzhe Yuan, Xiang Peng, Bin Gu, and Huan Xiong. Plug-and-play spiking operators: Breaking the nonlinearity bottleneck in spiking transformers. *arXiv preprint arXiv:2605.20289*, 2026.

Lusen Zhao, Zihan Huang, Jianhao Ding, and Zhaofei Yu. TTFSFormer: A TTFS-based lossless conversion of spiking transformer. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 77558–77571. PMLR, 13–19 Jul 2025.

Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

Rui-Jie Zhu, Qihang Zhao, Guoqi Li, and Jason K Eshraghian. SpikeGPT: Generative pre-trained language model with spiking neural networks. *arXiv preprint arXiv:2302.13939*, 2023.

A Experimental Details

A.1 Checkpoints and Data Populations

Table 5 lists the source checkpoints and data populations for the main conversion tables and the timing-noise comparison. All calibration populations are fixed seed-0 subsets of the corresponding training split.

Table 5: Models, evaluation populations, and calibration populations used for conversion and timing-noise evaluations.

(a) Models and checkpoints

Task	Architecture	Source checkpoint
CIFAR-10	CCT-7/3×1	`cct_7_3x1_32_cifar10_300epochs.pth`
CIFAR-10	ViT-S/16	`MF21377197/vit-small-patch16-224-finetuned-Cifar10`
ImageNet-1k	ViT-S/16	`timm/vit_small_patch16_224.augreg_in21k_ft_in1k`
ImageNet-1k	ViT-B/16	`timm/vit_base_patch16_224.augreg2_in21k_ft_in1k`
ImageNet-1k	ViT-L/16	`timm/vit_large_patch16_224.augreg_in21k_ft_in1k`
SST-2	RoBERTa-B	`Bhumika/roberta-base-finetuned-sst2`
SST-2	RoBERTa-L	`philschmid/roberta-large-sst2`
WikiText-2	GPT-2	`neulab/gpt2-finetuned-wikitext103`
WikiText-2	Llama 2 7B	`meta-llama/Llama-2-7b-hf`

(b) Evaluation settings

Task	Architecture	Evaluation	Calibration	Batch
CIFAR-10	CCT-7/3×1	test 10,000	train 5,000	8
CIFAR-10	ViT-S/16	test 10,000	train 5,000	32
ImageNet-1k	ViT-S/16	validation 5,000	train 5,000	32
ImageNet-1k	ViT-B/16	validation 5,000	train 5,000	32
ImageNet-1k	ViT-L/16	validation 5,000	train 5,000	32
SST-2	RoBERTa-B	validation 872	train 5,000	8
SST-2	RoBERTa-L	validation 872	train 5,000	8
WikiText-2	GPT-2	test 2,891 nonempty texts	train 5,000	8
WikiText-2	Llama 2 7B	test 2,891 nonempty texts	train 5,000	8

A.2 GELU and Calibration

ViT and RoBERTa ANN references retain their checkpoints' exact erf GELU, while the converted models use the cubic tanh GELU in Table 2. GPT-2 uses cubic tanh GELU in both models. Thus, the ViT and RoBERTa differences Δ include this GELU approximation and possible clamping effects. All converted models using GELU use the same f_{Pow} composition.

All reported ViT conversion rows use double precision with base time constants $\tau_s = \tau_m = 1$. Individual logarithmic ϕ operations use the ratios required by their compositions: $3\tau_m$ for the GELU cubic and $\tau_s/2$ for the LayerNorm variance. Analytic ranges are used directly; data-dependent ranges are fixed by layer-wise calibration as described in Section 5.1. Per-layer calibration uses two passes over 5,000 training examples: the first records extrema and the second fills 2,048-bin histograms. The retained quantiles are 0 and 1. For ranges symmetric about zero, the larger absolute observed endpoint determines both endpoints. When both endpoints are selected by calibration, each is extended by 5% of the selected interval width; analytically fixed bounds are specified separately. For LayerNorm/RMSNorm log and GELU cubic inputs, the positive floor is 10^{-5}, while the upper magnitude follows from the calibrated input range. The GELU exponential input potential z is

clamped to $[-80\tau_s, 80\tau_s]$. The reported evaluations use $\tau_s = 1$ for this gate, giving the numerical range $[-80, 80]$. This fixed limit is independent of layer-wise calibration; SwiGLU uses a separate limit (Appendix B.5).

A.3 CCT Evaluation and Software Environment

CCT uses the official 300-epoch CIFAR-10 checkpoint in Table 5, from Hassani et al. (2021) (`https://github.com/SHI-Labs/Compact-Transformers`), with double precision and the calibration procedure in Appendix A.2. Its learned convolution, affine, and normalization modules are checked for operator conversion before evaluation; learned sequence pooling and the classifier also use the proposed compositions.

The evaluation environment uses Python 3.12, PyTorch 2.11.0 with CUDA 12.8, Transformers 5.8.0.dev0, and timm 1.0.26. The reported runs use NVIDIA RTX A6000 GPUs.

B Operator Identities and Finite Domain Conditions

The derivations state the noise-free identities used by Table 2 under finite declared domains. They establish functional composition within those domains; they do not claim exactness of a physical circuit when clamping, a positive log input floor, a deadline miss, or analog non-idealities are active. Potentials and times below are normalized coordinates: the unit slope of ϕ_{NP} fixes the conversion of potential to latency, rather than equating physical voltage and time units.

B.1 ϕ Operators from Potential to Spike Time

For a potential $S \in [V_{\min}, V_{\max}]$, the ϕ_{NP} operator maps the upper endpoint to zero,

$$\phi_{\mathrm{NP}}(S) = V_{\max} - S, \qquad T_{\mathrm{NP}} = V_{\max} - V_{\min}. \tag{10}$$

The notation in Table 1 is the case $V_{\max} = \theta$. For a positive domain $V \in [V_{\min}, V_{\max}]$, $V_{\min} > 0$, the ϕ_{NL} operator is

$$\phi_{\mathrm{NL}}(V; \tau_s) = \tau_s \log \frac{V_{\max}}{V}, \qquad T_{\mathrm{NL}} = \tau_s \log \frac{V_{\max}}{V_{\min}}. \tag{11}$$

Equivalently, a fixed window T_{NL} represents values down to $V_{\min} = V_{\max} \exp(-T_{\mathrm{NL}}/\tau_s)$. Thus the positive lower endpoint is part of the operator domain rather than a value inferred after applying ϕ_{NL}.

B.2 Temporal Readout and Basic Compositions

The compositions use a drive that is constant during each pulse-width integration, in units where $I(v) = v$:

$$\psi_{\mathrm{Int}}(t_1, t_2; v) = v(t_2 - t_1). \tag{12}$$

The drive v may depend on the input; it is not necessarily a fixed synaptic weight. For a signed drive, define locally $v^+ = \max(v, 0)$ and $v^- = \max(-v, 0)$. Let d_j be the nonnegative duration from delivered event t_j to a shared observation deadline, and zero if that event is not delivered. Two nonnegative accumulator states give

$$Q^+ = v^+ d_1 + v^- d_2, \qquad Q^- = v^+ d_2 + v^- d_1, \qquad Q^+ - Q^- = v(d_1 - d_2). \tag{13}$$

When both events arrive by the deadline, this is $v(t_2 - t_1)$ for either temporal ordering. No integration backwards in time is required. There are four possible current connections but only two accumulators; this construction does not by itself specify circuit area or energy. The exponential-difference primitive directly maps two spike times as

$$\psi_{\mathrm{ED}}(t_1, t_2; \tau_m) = \exp\left(\frac{t_2 - t_1}{\tau_m}\right). \tag{14}$$

This mapping may alternatively be realized by composing fixed-gain ψ_{Int}, ϕ_{NP}, and ψ_{NE}. We treat this decomposition as a possible implementation rather than the definition of ψ_{ED}. For the numerical

evaluations, we use this decomposition; in the timing-noise experiments, its internal ϕ_{NP} operation is perturbed as well. For example, with both input times in $[0, T]$, the implemented decomposition is

$$\begin{aligned} p = \psi_{\mathrm{Int}}(t_1, t_2; -1) = t_1 - t_2, \qquad s = \phi_{\mathrm{NP}}(p) = T - p, \\ \psi_{\mathrm{ED}}(t_1, t_2; \tau_m) = e^{T/\tau_m} \psi_{\mathrm{NE}}(2T; s) = e^{-p/\tau_m}. \end{aligned} \tag{15}$$

Here $p \in [-T, T]$, $s \in [0, 2T]$, and $\psi_{\mathrm{NE}}(2T; s)$ denotes unit-gain exponential readout at local time $2T$ after a spike at s. The gain e^{T/τ_m} is fixed by the declared interval. This identity assumes delivery of the input and internal events; misses use the actual accumulator or reset state at the observation deadline.

For a bounded operand v_2, multiplication follows from the oriented pulse width. Both ϕ_{NP} operations use one local interval containing v_2 and zero:

$$\psi_{\mathrm{Int}}(\phi_{\mathrm{NP}}(v_2), \phi_{\mathrm{NP}}(0); v_1) = v_1[V_{\max} - (V_{\max} - v_2)] = v_1 v_2. \tag{16}$$

For division, let $0 < X \le Y$ share the same positive domain and logarithmic time constant. Their common offset cancels:

$$\psi_{\mathrm{ED}}(\phi_{\mathrm{NL}}(X), \phi_{\mathrm{NL}}(Y); \tau_s) = \exp\left(\frac{\tau_s \log(X/Y)}{\tau_s}\right) = \frac{X}{Y}. \tag{17}$$

The division operator f_{Div} consequently propagates the fixed range $[0, 1]$. Likewise, at a fixed observation time t_{obs}, composing $\phi_{\mathrm{NP}}(v) = \theta - v$ with ψ_{NE} gives $w \exp(-(t_{\mathrm{obs}} - \theta)/\tau_m) \exp(-v/\tau_m)$. We set $w = \exp((t_{\mathrm{obs}} - \theta)/\tau_m)$, so normalized decoding yields $f_{\mathrm{Exp}}(v) = \exp(-v/\tau_m)$. This normalization is part of f_{Exp} and therefore precedes the addition in $1 + f_{\mathrm{Exp}}(v)$ in the GELU composition. The mapping $\exp(-v)$ in Table 2 uses $\tau_m = 1$.

B.3 Power and Signed Cubic

For a positive exponent p, let $v > 0$ and a positive reference R share one logarithmic domain. Applying ϕ_{NL} to v and R with $p\tau_m$, then applying ψ_{ED} to their spike times with τ_m, gives

$$\psi_{\mathrm{ED}}(\phi_{\mathrm{NL}}(v; p\tau_m), \phi_{\mathrm{NL}}(R; p\tau_m); \tau_m) = \left(\frac{v}{R}\right)^p. \tag{18}$$

A fixed receiving gain R^p restores v^p. Table 2 uses $R = 1$; the signed cubic follows from the positive and negative branches in Section 5.2 with $p = 3$. Values below the fixed positive log input floor are suppressed, while values above the declared positive input endpoint are clamped.

GELU gate. Let $a = \sqrt{2/\pi}(v + 0.044715v^3)$. The GELU exponential stage uses $\tau_m = \tau_s$, so its normalized response is $\exp(-z/\tau_s)$. Since $\frac{1}{2}[1 + \tanh(a)] = [1 + \exp(-2a)]^{-1}$, the fixed $2\tau_s$ input gain gives $z = 2\tau_s a$. Hence $2\sqrt{2/\pi} = \sqrt{8/\pi}$ in Table 2. The coefficients 0.044715, $\sqrt{2/\pi}$, and $2\tau_s$ are fixed synaptic gains; the division result and the input are combined by the final variable product.

B.4 Softmax, LayerNorm, and RMSNorm Compositions

Let $r_{ij} = f_{\mathrm{SDP}}(\mathbf{q}_i, \mathbf{k}_j) = -\langle \mathbf{q}_i, \mathbf{k}_j \rangle / \sqrt{d_k}$. Applying the exponential and division compositions gives

$$e_{ij} = f_{\mathrm{Exp}}(r_{ij}) = \exp\left(\frac{\langle \mathbf{q}_i, \mathbf{k}_j \rangle}{\sqrt{d_k}}\right), \qquad a_{ij} = f_{\mathrm{Div}}\left(e_{ij}, \sum_k e_{ik}\right). \tag{19}$$

Because $0 < e_{ij} \le \sum_k e_{ik}$, the ordered positive-domain condition stated for the division operator is satisfied and $a_{ij} \in [0, 1]$.

For LayerNorm, let $\mu = D^{-1} \sum_i x_i$ and $\hat{x}_i = x_i - \mu$, with $\hat{x}_i^+ = \max(\hat{x}_i, 0)$, $\hat{x}_i^- = \max(-\hat{x}_i, 0)$, and

$$v = \frac{1}{D} \sum_i \left[(\hat{x}_i^+)^2 + (\hat{x}_i^-)^2\right] + \epsilon. \tag{20}$$

If $V_{\max}$ is the shared upper endpoint for $\hat{x}_i^+$ and $\hat{x}_i^-$, the spike times for the active branches and the variance are

$$t_i^{\pm} = \tau_s \log \frac{V_{\max}}{\hat{x}_i^{\pm}}, \qquad t_v = \frac{\tau_s}{2} \log \frac{V_{\max}^2}{v} = \tau_s \log \frac{V_{\max}}{\sqrt{v}}. \tag{21}$$

The exponential-difference primitive therefore gives

$$\psi_{\mathrm{ED}}(t_i^{\pm}, t_v; \tau_s) = \frac{\hat{x}_i^{\pm}}{\sqrt{v}}, \qquad y_i = \gamma_i \left[\psi_{\mathrm{ED}}(t_i^{+}, t_v; \tau_s) - \psi_{\mathrm{ED}}(t_i^{-}, t_v; \tau_s)\right] + \beta_i. \tag{22}$$

Thus LayerNorm uses two unrestricted ψ_{ED} branches rather than f_{Div}, and its normalized output is not restricted to $[0, 1]$.

These identities require finite declared bounds, a strictly positive shared domain for logarithmic operands, a common observation deadline for signed temporal differences, and representable exponential outputs. LayerNorm additionally uses explicit positive and negative branches, a positive log input floor, the variance epsilon, inactive branch masking, and output clamping. Its ideal normalization identity therefore applies only when these implementation constraints do not alter the operands or outputs.

Under the same finite-domain conditions, RMSNorm follows by omitting mean subtraction ($\hat{x}_i = x_i$), using $v = D^{-1} \sum_i x_i^2 + \epsilon$, and setting the output bias $\beta_i = 0$.

B.5 SwiGLU Composition

For gate and up-projection values u and v, respectively, SwiGLU is $vu\,\sigma(\beta u)$ elementwise, where $\sigma(x) = 1/(1 + e^{-x})$ in this subsection; Llama uses $\beta = 1$. Fix $\tau_s > 0$ and use $\tau_m = \tau_s$ for ψ_{NE}. For this construction only, let $z = \beta\tau_s u$ have declared interval $[a, b]$ with $a < b$, and let $T = b - a$. Before any finite-domain clamping, the composition is

$$E = e^{-a/\tau_s} \psi_{\mathrm{NE}}(T; \phi_{\mathrm{NP}}(z)) = e^{-\beta u}, \tag{23}$$

$$G = \psi_{\mathrm{ED}}(\phi_{\mathrm{NL}}(1; \tau_s), \phi_{\mathrm{NL}}(1 + E; \tau_s); \tau_s) = \frac{1}{1 + E}, \tag{24}$$

$$S = \psi_{\mathrm{Int}}(\phi_{\mathrm{NP}}(G), \phi_{\mathrm{NP}}(0); u) = uG, \tag{25}$$

$$f_{\mathrm{SwiGLU}}(u, v) = \psi_{\mathrm{Int}}(\phi_{\mathrm{NP}}(S), \phi_{\mathrm{NP}}(0); v) = vS = vu\,\sigma(\beta u). \tag{26}$$

Here $\phi_{\mathrm{NP}}(z) = b - z$ and the exponential is read at T; its fixed gain precedes the unit bias in $1 + E$, as in the exponential normalization above. The logarithmic operations share τ_s and $[1, 1 + e^{-a/\tau_s}]$, with no subtraction of the lower endpoint. The fixed scaling $\beta\tau_s$ preserves the activation when the time constant changes. Each final product uses an input-dependent drive and its own interval containing zero, with signed integration as in Equation (13).

Each stage starts after the preceding potential is available and uses its own local times. Potentials must be held and supplied as constant currents during the next integration stage; this remains an operator-level assumption.

These identities use the conditions of Appendix B. The SwiGLU exponential input $z = \beta\tau_s u$ is clamped to $[-20\tau_s, 20\tau_s]$, equivalently $z/\tau_s \in [-20, 20]$. GELU uses a separate limit (Appendix A). Subsequent potentials are clamped to their declared intervals. With active clamping, the equations use the clamped z and the input interval of its ϕ operation, so $E = e^{-\beta u}$ need not hold. Timing perturbations and missed events use the observation-time readout of Section 5.3, including the internal ϕ_{NP} operation in the implemented ψ_{ED} decomposition.

C Synaptic Operation Counts for ViT

This appendix derives the synaptic operation (SOP) estimates reported in Table 3. The derivation follows the Data SOP and Global SOP convention used throughout this work. It is an operation-count model for the stated TTFS mapping and does not represent a complete physical implementation.

C.1 Counting Convention and Notation

Data SOP counts a spike delivered to a destination. Global SOP counts synchronization of ϕ operations and scalar-reference delivery. Every ϕ operation receives one synchronization event; a scalar reference is synchronized once per operator call and delivered to each destination. Under timing noise, this scalar reference is sampled once and its perturbed timestamp is shared by all destinations in that

operator call. Bias, residual addition, mean computation, other potential sums, positional encoding, and class-token initialization incur no additional SOP under this abstraction. Both signed branches used by GELU and LayerNorm are counted. Fixed GELU coefficients are absorbed into synaptic weights, whereas the final product of two variable values is counted.

Let P be the number of image patches, $N = P + 1$ the token count including the class token, d_{in} the flattened patch dimension, D the embedding dimension, H the MLP hidden dimension, h the number of attention heads, L the number of Transformer blocks, and C the number of classes. For the models in Table 3, $P = 196$, $N = 197$, $d_{\text{in}} = 768$, and $H = 4D$. All counts are per image.

For R vectors of I input features and O output features, an affine projection contributes

$$\begin{aligned} \mathrm{SOP}_{\text{data}}^{\text{linear}} &= RIO, \\ \mathrm{SOP}_{\text{global}}^{\text{linear}} &= RI + RO + 1. \end{aligned} \tag{27}$$

The terms in the second line count input synchronization, one scalar-reference synchronization, and one reference delivery to each output.

C.2 GELU and LayerNorm

Table 6 expands one GELU call over N tokens and H hidden features. The signed cubic contains separate ϕ operations for $\max(v, 0)$ and $\max(-v, 0)$ and two internal ϕ operations. Division includes the explicit ϕ operation for the tensor of ones, the ϕ operation for the denominator, and the internal ϕ operation. The fixed gains in the GELU gate contribute no additional multiplication cost under the counting convention above.

Table 6: SOP derivation for one GELU call.

Component	Data SOP	Global SOP	Total SOP
Signed cubic	$4NH$	$6NH + 1$	$10NH + 1$
Exponential	NH	NH	$2NH$
Division	$3NH$	$3NH$	$6NH$
Final variable product	NH	$2NH + 1$	$3NH + 1$
GELU total	$9NH$	$12NH + 2$	$21NH + 2$

For one LayerNorm (LN) over $N \times D$ features, the squared positive and negative branches use the composed multiplication operator. Each variance event fans out to both signed feature arrays, and both exponential-difference branches retain their internal ϕ operation. The learned scale γ undergoes one ϕ operation per feature and is delivered to all N tokens. Table 7 gives the component costs and their total.

Table 7: SOP derivation for one LayerNorm.

Component	Data SOP	Global SOP	Total SOP
Positive and negative branch squares	$2ND$	$4ND + 2$	$6ND + 2$
Logarithmic ϕ operations and exponential difference	$6ND$	$4ND + N$	$10ND + N$
Learned-scale ϕ operation and product	ND	$ND + D + 1$	$2ND + D + 1$
LayerNorm total	$9ND$	$9ND + D + N + 3$	$18ND + D + N + 3$

C.3 Multi-Head Self-Attention

Table 8 includes the query, key, and value projections, scores, softmax over all h heads, the weighted value sum, and the attention output projection.

Table 8: SOP derivation for one multi-head self-attention module.

Component	Data SOP	Global SOP
Query, key, and value projections	$3ND^2$	$6ND + 3$
Attention scores	N^2D	$ND + hN^2 + 1$
Exponential	hN^2	hN^2
Division	$3hN^2$	$2hN^2 + hN$
Weighted value sum	N^2D	$2ND + 1$
Attention output projection	ND^2	$2ND + 1$
Total	$4ND^2 + 2N^2D + 4hN^2$	$11ND + 4hN^2 + hN + 6$

C.4 MLP and One Transformer Block

Applying Eq. 27 to the two MLP projections and adding the GELU cost gives

$$\begin{aligned} \mathrm{SOP}^{\mathrm{MLP}}_{\mathrm{data}} &= 2NDH + 9NH, \\ \mathrm{SOP}^{\mathrm{MLP}}_{\mathrm{global}} &= 2ND + 14NH + 4, \\ \mathrm{SOP}^{\mathrm{MLP}} &= 2NDH + 2ND + 23NH + 4. \end{aligned} \tag{28}$$

A Transformer block combines two LayerNorm modules, one attention module, and one MLP; residual sums follow the counting convention above. Summing Tables 7 and 8 with Eq. 28 gives

$$\mathrm{SOP}^{\mathrm{block}} = 2\,\mathrm{SOP}^{\mathrm{LN}} + \mathrm{SOP}^{\mathrm{attn}} + \mathrm{SOP}^{\mathrm{MLP}}. \tag{29}$$

Data SOP and Global SOP follow the same component sum. For the $H = 4D$ configuration used here, the total is

$$\mathrm{SOP}^{\mathrm{block}} = 12ND^2 + 2N^2D + 8hN^2 + 141ND + 2D + (h+2)N + 16. \tag{30}$$

C.5 End-to-End ViT and Numerical Values

Patch embedding processes the P image patches, whereas every Transformer block processes all $N = P + 1$ tokens. The stem, final LayerNorm, and TTFS classification head contribute

$$\begin{aligned} \mathrm{SOP}^{\mathrm{stem}} &= Pd_{\mathrm{in}}D + Pd_{\mathrm{in}} + PD + 1, \\ \mathrm{SOP}^{\text{final LN}} &= 18ND + D + N + 3, \\ \mathrm{SOP}^{\mathrm{head}} &= DC + D + C + 1. \end{aligned} \tag{31}$$

The end-to-end count is therefore

$$\boxed{\mathrm{SOP}^{\mathrm{ViT}} = \mathrm{SOP}^{\mathrm{stem}} + L\,\mathrm{SOP}^{\mathrm{block}} + \mathrm{SOP}^{\text{final LN}} + \mathrm{SOP}^{\mathrm{head}}.} \tag{32}$$

Table 9 substitutes the checkpoint dimensions into Eq. 32. ViT-S/B/L use $(D, h, L) = (384, 6, 12)$, $(768, 12, 12)$, and $(1024, 16, 24)$, respectively. The class count is $C = 10$ for CIFAR-10 and $C = 1000$ for ImageNet-1k.

Table 9: Numerical SOP estimates used in Table 3.

Dataset	Architecture	Total SOP
CIFAR-10	ViT-S/16	4,750,469,300
ImageNet-1k	ViT-S/16	4,750,850,450
ImageNet-1k	ViT-B/16	17,867,607,866
ImageNet-1k	ViT-L/16	62,360,701,602

The classifier counted in Eq. 31 is the evaluated TTFS linear composition, receiving the final LayerNorm output and its declared potential range to produce logits.

D Additional Model Evaluations

This section gives the settings and additional results for discrete-time simulation and timing-noise evaluations. Checkpoints and calibration settings are listed in Appendix A; the Llama timing-noise diagnostic uses the smaller populations specified in Appendix D.4. In the timing-noise experiments below, calibration and evaluation populations remain fixed across conditions. Each stochastic condition uses seeds $\{0, 1, 2\}$; results report means and 95% Student-t intervals. The observation margin is $4\sigma_t$ except when varied explicitly. The ViT-B/16 timing-noise experiments use double precision and the fixed first 5,000 ImageNet-1k validation images.

D.1 Discrete-time Simulation

Figure 4 shows the steps per ϕ operator window at the global interval $\Delta t = 0.01$ used in Figure 2. The single ViT-B/16 forward pass uses the same source revision, checkpoint, preprocessing, and frozen calibration as that 500-image accuracy sweep. Across its 651 windows, the median is 3,248 steps, the mean is 6,747.16, the population variance is 9.16×10^7 steps^2, and the range is 100–105,331 steps. Each application contributes one count regardless of tensor size or event count; this is a distribution of simulation steps from one forward pass.

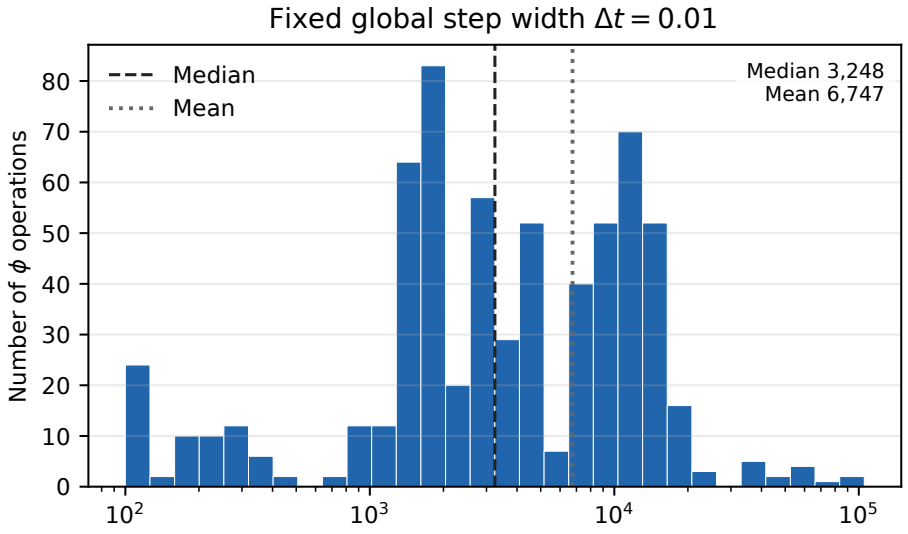


Figure 4: Simulation time steps per window of a ϕ operator with a global interval of $\Delta t = 0.01$. The 651 windows come from one ViT-B/16 forward pass with the frozen calibration used in Figure 2; each window contributes one count.

D.2 Model Evaluations Using Measured Timing Variation

The evaluations in Figure 3 use the timing variation measured with the protocol in Appendix E.

Noise model and scope. The simulations apply Equation 9 to one spike per value at each ϕ operation. The measured scale includes synaptic loading (Appendix E), whose contribution is included in the timing perturbation. Other Ψ operations and stored potentials are ideal. The model excludes additional spikes, correlations across events and primitives, routing faults or spike loss, static device mismatch, and temperature variation. There is no training or calibration on hardware. Empirical residuals, deterministic mapping offsets, and hardware miss probabilities are not injected; misses occur when sampled times exceed the simulated observation deadline.

Model comparison. The model comparison uses the populations in Table 5 and the two measured fractions in Table 11. Equation 9 scales them by a common α; $\alpha = 1$ uses the measured scales while preserving their ratio. For GPT-2, inverse perplexity is computed before aggregation across seeds.

Cumulative block experiment. The cumulative block experiment applies $\alpha \in \{0.03, 0.05, 0.075, 0.1\}$ to the first $K \in \{0, 1, 2, 4, 8, 12\}$ encoder blocks of ViT-B/16, perturbing both ϕ operators in those blocks. Embeddings, final normalization, and the classifier receive no injected noise. The $K = 0$ run supplies the shared deterministic reference.

Detailed results. At $\alpha = 1$, the model comparison gives accuracies of 10.18% for CCT-7, 0.11% for ViT-S/16, 0.09% for ViT-B/16, and 50.27% for RoBERTa-B. In addition to the cumulative endpoints reported in Section 6.4, $\alpha = 0.075$ gives 67.51%, 65.45%, and 61.84% at $K = 2, 4, 8$. At $K = 12$, $\alpha = 0.05$ retains 70.81% and $\alpha = 0.1$ reaches 14.55%.

Linear operations. At $\alpha = 0.075$, we keep timing noise active throughout ViT-B/16 and disable it in one group of linear operations at a time. The condition with noise throughout includes the patch projection, final LayerNorm, and classifier, which are excluded from the cumulative block experiment. The clean reference is 86.00%, and the mean with noise throughout is 45.39%. Relative to the latter, disabling noise in the MLP linear projections raises mean top-1 accuracy to 74.83%, a paired change of 29.43 ± 3.55 percentage points (paired mean and 95% Student-t interval). This is the largest recovery among the five groups of linear operations; Q/K/V projections give 4.17 ± 1.94 points. Each remaining group changes accuracy by at most 0.53 points in magnitude. The differences are not an additive decomposition of the total accuracy loss.

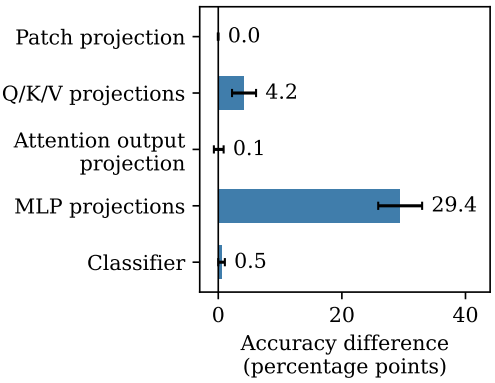


Figure 5: Change in ViT-B/16 top-1 accuracy after disabling timing noise in one group of linear operations at $\alpha = 0.075$, relative to the condition with noise throughout the network. Bars show paired mean differences and 95% Student-t intervals over three seeds on the fixed ImageNet-1k validation 5k.

D.3 Timing Noise and Observation Margin Sweeps

This ViT-B/16 sweep uses batch size 32. For each ϕ operation with window T, it sets $\sigma_t = r_t T$ with the same r_t for NP and NL. The timing-noise grid is $r_t = 10^{-5} 10^{i/8}$, $i = 0, \ldots, 8$, at margin $4\sigma_t$. The margin grid is $k \in \{0, 0.5, 1, 1.5, 2, 2.5, 3, 4, 5, 6, 8, 10, 12\}$ with margin $k\sigma_t$ at $r_t = 10^{-5}$. Deadline-miss rates pool event counts across the three seeds. Analytic operator limits remain fixed; delivered spikes retain their sampled timestamps, including arrivals within the added margin. The internal ϕ_{NP} operation in exponential difference also receives noise, and fixed GELU coefficients use constant synaptic scaling. These computational Gaussian sweeps use no BrainScaleS-2 measurements: Figure 3(a) uses distinct measured fractions, while panel (b) varies the number of noisy blocks at fixed margin.

Across the timing-noise grid, mean top-1 accuracy changes from 86.0267% at $r_t = 10^{-5}$ to 85.5133% at $r_t = 10^{-4}$. At $r_t = 10^{-5}$, increasing k from 0 to 4 changes mean accuracy from 77.3267% to 86.0267% and reduces the pooled deadline-miss rate from 6.9153% to 0.000423%.

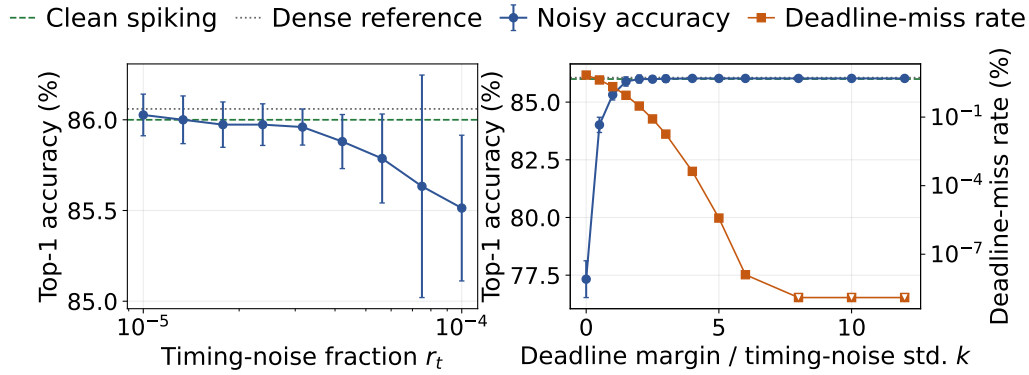


Figure 6: Simulated timing-noise robustness of ViT-B/16 on the fixed ImageNet-1k validation 5k. Left: top-1 accuracy as timing noise increases at deadline margin $4\sigma_t$. Right: top-1 accuracy and pooled deadline-miss rate as k changes at $r_t = 10^{-5}$. Error bars show 95% Student-t intervals over seeds $\{0, 1, 2\}$. Zero observed deadline-miss rates are shown at the plotting floor. These are computational Gaussian timing-noise results, not chip measurements.

D.4 Llama 2 7B Evaluation under Timing Noise

We evaluate the converted Llama 2 7B Transformers checkpoint `meta-llama/Llama-2-7b-hf` on WikiText-2 and IMDb as causal language modeling tasks. Perplexity on IMDb is not sentiment classification accuracy. This bounded diagnostic uses double precision, batch size 8, at most 128 tokens per text, and the first 128 test texts from each dataset. A shared layer-wise calibration is frozen after two passes over 32 WikiText-2 training texts. These reduced populations are not the full conversion or timing-noise experiments.

The noise-free converted model supplies the deterministic reference. Noisy conditions use Equation 9 with $\alpha \in \{10^{-5}, 10^{-4}, 10^{-3}\}$ and the common seeds and observation margin above. Table 10 reports mean perplexity and 95% Student-t confidence intervals; the deterministic reference is evaluated once. At multipliers 10^{-5} and 10^{-4}, the mean perplexity differs from the deterministic reference by at most 1.5% on both datasets. At 10^{-3}, the mean perplexity exceeds 500 on both datasets. The confidence intervals describe variation across three simulated timing noise draws for these fixed test texts; they do not include variation from selecting different texts or from physical devices.

Table 10: Llama 2 7B diagnostic perplexity on the first 128 test texts per dataset after calibration on 32 WikiText-2 training texts. The three noisy rows scale both measured timing noise fractions by the indicated multiplier; smaller perplexity is better. The noisy cells show mean perplexity and 95% Student-t confidence intervals over seeds $\{0, 1, 2\}$.

Condition	WikiText-2	IMDb
Source clean	15.184	13.791
Converted clean	15.183	13.791
10^{-5}	15.179 ± 0.004	13.791 ± 0.002
10^{-4}	15.401 ± 0.352	13.926 ± 0.211
10^{-3}	620.007 ± 351.519	525.515 ± 284.373

E BrainScaleS-2 Measurement Protocol for Timing Noise

We characterize two physical paths implementing ϕ operators on BrainScaleS-2. Each path uses synaptic loading to establish a membrane potential determined by the input code and then uses either ϕ_{NP} or ϕ_{NL} to produce a spike. The loading uses simultaneous input events with code-dependent synaptic weight and is not a pulse-width realization of ψ_{Int}. Measurements from one chip and calibration estimate timing variation in each path at separate operating points and latency budgets. They do not test the full converted Transformer on hardware or cover every fitted code. They combine

loading and ϕ operator errors and do not characterize other Ψ operations or correlations between primitives. For both paths, we record raw first-spike times without measuring the loaded membrane potential. The refractory interval extends beyond the observation deadline; an absent spike is a miss and is not assigned the deadline as its timestamp.

E.1 Physical Implementation of the ϕ_{NP} Operator

For the measurements used in the model evaluation, each trial begins with reset of the selected circuit. After release from the common reset state, 15 input channels emit simultaneous events before $t_{\mathrm{ref}} = 5\,\mu$s, with integer synaptic weights increasing with the potential code $q \in \{0, \ldots, 31\}$. A synchronized constant current then raises the loaded potential approximately linearly toward the fixed threshold. It runs from 5 to 91 μs with an observation deadline of 300 μs. The first-spike time relative to t_{ref} is the physical output. Code 31 is an immediate-firing endpoint diagnostic excluded from the fitted response and timing variation estimate.

In a separate direct-initialization diagnostic, q selects the reset and leak-potential settings, so releasing reset establishes the initial potential. The fixed-latency condition runs the constant current from t_{ref} until threshold crossing or 25 μs, with a 60 μs observation deadline.

E.2 Physical Implementation of the ϕ_{NL} Operator

The ϕ_{NL} operator uses the same type of reset and synaptic-loading path, with positive codes $q \in \{1, \ldots, 31\}$. After the code-dependent initial membrane potential has been loaded, a fixed set of simultaneous excitatory input events is delivered at t_{ref}. Their fixed synaptic weight and number of input channels generate the same exponentially decaying synaptic current for every code. No constant-current interval is used in this measurement. The code-dependent state changes the remaining distance to the fixed threshold, and the first threshold-crossing time realizes the logarithmic response fitted in Appendix E.3 on its positive domain. For each measurement, we record the circuit coordinate, neuron and synaptic parameter settings, and calibration identity.

E.3 Timing Variation and Normalization

Conditional timing variation. Let $t_{iqn}^{(e)}$ denote the first spike time for ϕ operator $e \in \{\mathrm{NP}, \mathrm{NL}\}$, physical circuit i, potential code q, and repetition n. Let $\mathcal{Q}_e$ denote the measured fitted codes for ϕ operator e: here $\mathcal{Q}_{\mathrm{NP}} = \{0, 15, 30\}$ and $\mathcal{Q}_{\mathrm{NL}} = \{1, 8, 31\}$. A sample is usable only when a spike is delivered by the observation deadline, its timestamp is finite, and the observation is not saturated. For each code, we compute the unbiased sample variance s_{eiq}^2 of usable first spike times. The code-uniform scale is

$$\sigma_{t,e,i}^{\mathrm{unif}} = \left(\frac{1}{|\mathcal{Q}_e|} \sum_{q \in \mathcal{Q}_e} s_{eiq}^2 \right)^{1/2}. \tag{33}$$

It is the square root of a code-uniform average of conditional variances, not the standard deviation of all raw spike times. Code means therefore remain part of the transfer function rather than being counted as timing variation. Deadline misses are not replaced by the deadline when computing timing variation.

Primitive-specific normalization. Calibration repetitions fit the synaptically loaded ϕ_{NP} response as $\widehat{f}_{\mathrm{NP},i}(q) = a_i + b_i q$. They fit the logarithmic response as $\widehat{f}_{\mathrm{NL},i}(q) = c_i - \tau_{\mathrm{eff},i} \log(q - q_{\mathrm{lb},i})$ with $q > q_{\mathrm{lb},i}$. The parameters c_i, $\tau_{\mathrm{eff},i}$, and $q_{\mathrm{lb},i}$ are fitted separately for each circuit. We define separate signal spans fixed by calibration,

$$\widehat{T}_{\mathrm{NP},i} = |\widehat{f}_{\mathrm{NP},i}(31) - \widehat{f}_{\mathrm{NP},i}(0)| = 31|b_i|, \tag{34}$$

$$\widehat{T}_{\mathrm{NL},i} = |\widehat{f}_{\mathrm{NL},i}(31) - \widehat{f}_{\mathrm{NL},i}(1)|, \tag{35}$$

and normalize each ϕ operator by its own span. For the reported validation values, $\sigma_{t,e,i} = \sigma_{t,e,i}^{\mathrm{unif}}$ is computed from the held-out repetitions, giving

$$r_{t,i}^{(e)} = \frac{\sigma_{t,e,i}}{\widehat{T}_{e,i}}. \tag{36}$$

The transfer parameters and spans are frozen after calibration and reused for the disjoint validation repetitions. Table 11 reports the validation timing scale, fitted signal span, normalized timing variation, and time budgets of the two circuits used in the model evaluations. Although NP code 31 is excluded from the fit and variance estimate (Appendix E.1), $31|b_i|$ extrapolates the fitted response to the full 32-code interval as predeclared.

Table 11: Timing measurements for the selected NP and NL circuits. $\sigma_{t,e,i}$ is estimated from validation repetitions, while $\widehat{T}_{e,i}$ is fixed by calibration. Time entries are in microseconds; the window endpoints and observation deadline use the same measurement time origin.

Operator	Circuit	$\sigma_{t,e,i}$	$\widehat{T}_{e,i}$	$r_{t,i}^{(e)}$	Window	Deadline
ϕ_{NP}	184	0.200021	18.455333	0.010838111	5–91	300
ϕ_{NL}	1	0.0770333	3.129226	0.024617377	5–25	60

Mapping error and the 32-code grid. For every fitted code, the deterministic mapping error is the difference between its calibration mean and the frozen transfer curve. It is retained as a code-indexed offset and is not merged into the within-code timing variance. The 32 potential codes form the physical measurement grid. The model experiments use continuous potentials and inject timing noise only; they do not apply 32-level quantization or deterministic mapping offsets.

E.4 Circuit Selection and Validation

Measured population. We obtained measurements from 63 of 64 requested ϕ_{NP} circuit coordinates; four meet the revised NP selection criteria below. Of 512 requested ϕ_{NL} coordinates, 58 meet the NL criteria. The reported medians describe only these selected circuits, not the full chip. Within each eligible circuit set, we select a circuit from the middle of the calibration r_t ranking and report its held-out validation scale. We call these two distinct circuits the screening median pair; their coordinates are 184 (NP) and 1 (NL). The model comparison and cumulative block experiment use this pair, without sampling variation across a circuit population.

The measured representative codes are the sets $\mathcal{Q}_e$ defined above. Each NP code has eight calibration and eight validation repetitions; each NL code has 16 of each. Calibration precedes validation, and we record the acquisition order and wall-clock timestamps. A full validation would cover all codes $0, \ldots, 30$ for NP and $1, \ldots, 31$ for NL, with 128 calibration and 128 validation repetitions per code. This full validation has not been completed; its data do not contribute to the reported r_t values.

Selection criteria. We initially specified the same selection criteria for both ϕ operators: at least two usable first-spike times at every fitted code, the declared direction of the relation between code and spike time, a deadline-miss rate of at most 5%, a fraction of trials with more than one spike of at most 0.1%, and a saturation rate of at most 1%. When a trial produces multiple spikes, only the first enters the timing statistic. For NL, the deadline-miss limit restricts the timing variation estimate to circuits that usually produce a spike; we applied these criteria at every required measurement stage. Too few NP circuits met the same criteria to summarize their variation. After inspecting the measurements, we therefore relaxed the NP criteria to require finite conditional variance at each fitted code, a finite fitted span, and a fraction of trials with more than one spike of at most 0.1%. Missed deadlines and the measured relation between code and spike time do not determine NP circuit selection. Thus, the four selected NP circuits, including median coordinate 184, provide an exploratory estimate. They do not establish a validated NP response or represent the full chip.

Full validation would additionally require root-mean-square error of the fitted response at most 5% of the signal span, rank correlation at most -0.95 between NP code and mean spike time (the expected decreasing relation), at most 10% relative change between the calibration and validation-only fits, and membrane observations for synaptic loading. The current measurements do not establish these conditions or the prerequisite based on direct initialization.

E.5 Observation Deadlines and Latency

For the selected NP circuit 184, no deadline misses were observed in the validation repetitions (0%). A non-firing trial that is not explained by a late first spike is not identified with the Gaussian deadline tail.

Latency budgets. Table 11 gives the operating windows and observation deadlines. NL uses the time budget of the fixed-latency diagnostic in Appendix E.1; NP uses the longer operating point. We record the ϕ operator window, observation time, deadline, and the deadline measured from primitive start as the maximum declared primitive latency. The two conditions are not ranked using r_t alone.

The use of these measurements in the model evaluations is described in Appendix D.2.