READING AND WRITING RESIDUAL STATES FOR DIFFUSION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion Transformers have advanced mainly through improved backbones, representations, and objectives, while the residual pathway that carries information across depth remains largely fixed. This work investigates an orthogonal question, *i.e., how should a denoiser read from and write to multiple persistent residual states?* We present a systematic study of this design in diffusion Transformers, using Gated Residual as a starting point. Controlled experiments show that adaptive access improves generation and that the scale of the mixed feature substantially affects its effectiveness. We introduce a direction-calibrated reader that preserves the direction selected by adaptive gates and matches the mixture's magnitude to an ungated average of normalized states. For writing, we find that modulating the update component orthogonal to each stored state outperforms modulating the entire update. This motivates a noise-conditioned tangent writer that controls this component according to the noise level. Both designs retain a single forward of each attention and MLP sublayer across states. The resulting method improves both pixel- and latent-space diffusion models and remains compatible with representation alignment methods. Code will be made publicly available.

1 INTRODUCTION

Diffusion Transformers (DiT) have emerged as a foundation for visual generation, progressing from class-conditional image synthesis to high-resolution, large-scale text-to-image modeling (Bao et al., 2023; Peebles & Xie, 2023; Gao et al., 2023; Chen et al., 2025; Esser et al., 2024). Recent advances have refined different components of this paradigm: multimodal DiTs strengthen text–image interaction (Esser et al., 2024); compressed latent representations and efficient attention reduce the computational cost (Chen et al., 2025; Xie et al., 2025); guidance from external representations makes denoising features easier to learn (Yu et al., 2025; Leng et al., 2025; Shin et al., 2026; Wang et al., 2026; Singh et al., 2026); generative formulations reconsider the model's training objective (Ma et al., 2024; Li & He, 2026).

Cross-layer information flow provides another avenue for improving denoising. U-ViT took a pioneering step with long skip connections between shallow and deep layers (Bao et al., 2023), and PixelU further explores this for pixel-space generation (Guo et al., 2026). These designs prescribe which earlier layers each block can directly access and retain their activations for later fusion. Diffusion-Adaptive Routing instead learns to aggregate historical sublayer outputs under timestep conditioning (Xu et al., 2026). We investigate a different organization of cross-layer information: a fixed number of persistent residual states that each block adaptively reads and continually updates.

We systematically study how diffusion Transformers should select information from multiple residual states and incorporate new sublayer outputs into them. Each attention or MLP sublayer processes a single feature assembled from the residual states, and its output is then written back to those states (Zhu et al., 2025; Qwen Team, 2026). We use Gated Residual (GR), which implements adaptive reads and writes and has shown benefits in language modeling (Qwen Team, 2026), as our starting point. Controlled experiments on JiT-B show that adaptive reading improves generation and that branch-specific writing provides further gains (Table 1).

Reading selects the information passed from the states to a shared sublayer. Adaptive mixing changes both the direction and magnitude of this feature, which enters the sublayer without another normalization. Experiments show that readout scale affects generation and that retaining adaptive

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.

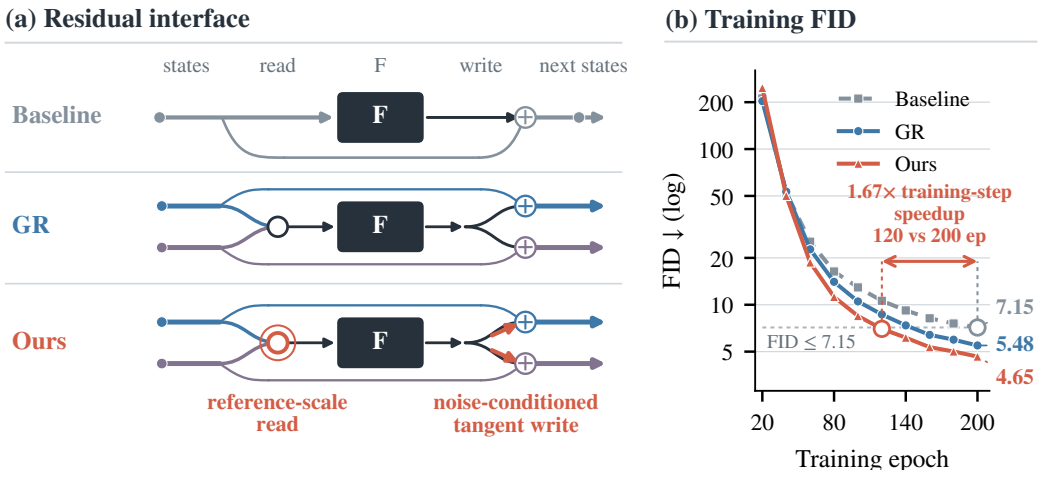


Figure 1: **Residual access and JiT training.** (a) Baseline uses one residual state; GR maintains multiple states with gated read/write around F. Ours calibrates the read to the reference scale and adds a noise-conditioned tangent correction to gated writes. (b) JiT-B/16 FID-50K (Heun50, CFG $= 2.3$). Ours matches baseline's 200-epoch FID by 120 epochs, a $1.67\times$ training-step speedup.

direction is more effective than retaining adaptive magnitude alone. Our *direction-calibrated reader* therefore preserves the gated direction and matches its RMS magnitude to that of the equal-weight average of normalized states. This reference represents the ungated readout of the same states and varies with their features. Writing incorporates each sublayer's output into the stored states, shaping the information available to subsequent blocks. We compare modulating the entire update with modulating its components parallel or orthogonal to each destination state. Selective modulation of the orthogonal, or tangent, component improves generation over whole-update scaling, and conditioning its strength on the noise level further improves over static tangent modulation. These findings motivate our *noise-conditioned tangent writer*, which adjusts the tangent component according to the noise level while preserving radial update. Figure 2 illustrates our choice. It improves ImageNet-256 generation across multiple backbones across pixel- and latent-space diffusion (Section 5.2).

Our contributions are threefold:

- We present a systematic study of multi-state residual connections in diffusion Transformers, examining how adaptive reading and writing affect generation.
- Based on controlled comparisons of readout scale and update geometry, we develop a direction-calibrated reader and a noise-conditioned tangent writer.
- We demonstrate improved generation across pixel- and latent-space diffusion backbones, with gains extending to models trained with representation alignment.

2 Related Work

Diffusion architectures and feature propagation. Diffusion Transformer research has developed scalable denoisers under diffusion and interpolant objectives (Peebles & Xie, 2023; Ma et al., 2024), multimodal architectures for text-conditioned generation (Esser et al., 2024), and efficient representations and attention for high-resolution synthesis (Xie et al., 2025). REPA improves training through representation alignment (Yu et al., 2025), while JiT revisits clean-image prediction in pixel space (Li & He, 2026). For cross-layer propagation, U-ViT and PixelU use long skip connections (Bao et al., 2023; Guo et al., 2026). DAR analyzes information flow along depth and denoising time and introduces timestep-adaptive aggregation over historical sublayer outputs (Xu et al., 2026). Our study focuses on adaptive reading and writing over a fixed number of continually updated residual states. It changes cross-layer information flow while sharing sublayer computation.

Residual connectivity and adaptive state access. Residual networks establish identity shortcuts across depth (He et al., 2016); ReZero and DeepNet improve signal propagation through residual

gating, scaling, and initialization (Bachlechner et al., 2021; Wang et al., 2024). Hyper-Connections expands the residual state while retaining shared sublayer computation, with learned aggregation, distribution, and inter-stream mixing (Zhu et al., 2025). Its DiT experiment validates a static two-stream variant for image generation. Manifold-Constrained Hyper-Connections (mHC) constrains inter-stream mixing toward a doubly stochastic matrix (Xie et al., 2026). GR uses independently normalized branches, element-wise read gates, and branch-specific write gates (Qwen Team, 2026). Attention Residuals aggregates preceding sublayer outputs or block-level summaries (Team et al., 2026). We build on GR to establish its effectiveness in diffusion and systematically examine the read/write design choices that affect denoising. This extends the application evidence from static Hyper-Connections to a dedicated study of adaptive state access and noise-conditioned updates.

Normalization and directional updates. SiameseNorm couples Pre-Norm-like and Post-Norm-like streams through shared residual blocks (Li et al., 2026). Its design reconciles normalization placement and gradient propagation. nGPT uses normalized representations and weight vectors, with learned interpolation and hypersphere normalization (Loshchilov et al., 2025). We leave the persistent-state norms unconstrained and instead design updates for subsequent normalized reads. The direction reader calibrates the mixed feature's magnitude to an ungated reference, while the noise-conditioned tangent writer modulates the update component orthogonal to each stored state.

3 Gated Residual for diffusion Transformers

Gated Residual (GR) (Qwen Team, 2026) maintains K persistent width-D states and uses input-dependent gates to read from and write to them. Each attention or MLP sublayer receives one width-D feature assembled from these states and returns a shared update.

Normalized reading. Let $H = \{h_i\}_{i=1}^{K}$ denote the residual states for one token before a sublayer, with $h_i \in \mathbb{R}^D$. At the network input, all states are initialized from the same patch embedding. The reader applies RMS normalization (Zhang & Sennrich, 2019) to each state independently and concatenates the resulting features:

$$z_i = \gamma_i \odot \frac{h_i}{\sqrt{D^{-1}\|h_i\|^2 + \epsilon}}, \qquad z = [z_1; \dots; z_K] \in \mathbb{R}^{KD}. \tag{1}$$

It then predicts an element-wise read gate $g_i \in (0,1)^D$ for each state and averages the gated features:

$$[g_1; \dots; g_K] = \mathrm{sigmoid}\big(W_{\mathrm{up}}\,\mathrm{SiLU}\big(\tfrac{1}{K}W_{\mathrm{down}}z\big)\big), \qquad r_g = \frac{1}{K}\sum_{i=1}^{K} g_i \odot z_i. \tag{2}$$

Here $W_{\mathrm{down}} \in \mathbb{R}^{k\times KD}$ and $W_{\mathrm{up}} \in \mathbb{R}^{KD\times k}$ form the gate network with bottleneck width k. Branch normalization replaces the backbone's original pre-sublayer normalization. The mixed readout receives adaptive backbone conditioning and enters attention or MLP without another normalization. The shared sublayer computes $u = F_\ell(r_g; t, c) \in \mathbb{R}^D$, including its output gate.

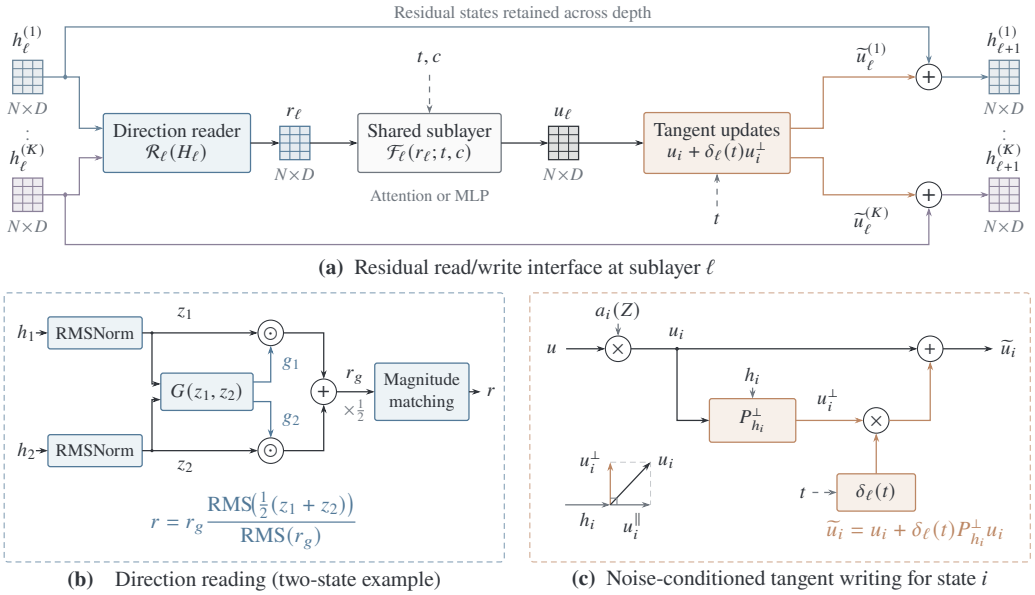
Adaptive writing. From the concatenated normalized features, the writer predicts a scalar injection weight per token and state as follows:

$$[a_1; \dots; a_K] = 2\,\mathrm{sigmoid}\big(\tfrac{1}{K}W_{\mathrm{inj}}z\big), \qquad h_i' = h_i + a_i u, \quad i = 1, \dots, K, \tag{3}$$

where $W_{\mathrm{inj}} \in \mathbb{R}^{K\times KD}$ and $a_i \in (0,2)$. Each attention and MLP sublayer has its own reader and writer. A final reader aggregates all K states for the prediction head. The states persist across depth, while every sublayer is evaluated once at width D.

4 Designing residual reading and writing for diffusion

We develop the read/write design through a controlled study of GR in diffusion. We first test whether adaptive access improves generation and where it should be placed. We then examine readout scale and direction to construct a calibrated reader, and compare update components and noise conditioning to construct a tangent writer. We then combine these choices into one interface.



(a) Residual read/write interface at sublayer ℓ

(b) Direction reading (two-state example)

(c) Noise-conditioned tangent writing for state i

Figure 2: **Residual-state reading and writing.** (a) The reader combines K states into one shared sublayer input; the writer distributes the output to the states. (b) Direction reading calibrates the adaptive mixture to the RMS magnitude of an equal-weight reference. (c) The noise-conditioned tangent writer modulates the component of each gated update orthogonal to its destination state.

Experimental setting. We use JiT-B/16 on ImageNet-256 (Russakovsky et al., 2015) and a patch-embedding bottleneck of 256. We train each model for 200 epochs with a global batch size of 1,024. We report FID (Heusel et al., 2017) (↓) on 50K samples generated from EMA checkpoints (decay 0.9999), using Heun50, CFG= 2.3 (Ho & Salimans, 2022). We use a state budget of $K = 2$ for these controlled comparisons; Appendix C.1 examines generation quality as the number of states increases. Baseline denotes the original single-state backbone shown in Figure 1. The adaptive-access experiments use original GR. Subsequent comparisons examine changes to this reader and writer, with the reference configuration specified in each subsection.

4.1 Adaptive Residual Access

Adaptive access or additional storage? GR improves FID from 7.15 to 5.48. This gain could reflect adaptive feature access, the accumulation of separate histories, or both. To distinguish these effects, we restrict the read and write gates while retaining the two state tensors (Table 1). Read-only GR uses adaptive reads and fixed unit writes. Write-only GR fixes the read gates at $1/2$ and retains adaptive branch writes. A shared-write control keeps input-dependent injection weights but averages them across branches before writing. Both restricted adaptive interfaces improve over Baseline, with read-only outperforming write-only. Moreover, read-only and shared-write GR improve FID even though their raw branch states remain identical. These models still expose the states through independent normalization gains and adaptive readers. Full GR further improves on read-only (5.97) and shared-write (6.09). The gain is therefore not explained by divergent storage alone: adaptive access already helps, and branch-specific writing adds to that benefit.

Finding 1. *How the residual states are accessed matters in addition to what they store. Adaptive reading and branch-specific writing together provide the strongest GR interface.*

Where should adaptive state access be placed? We next ask which sublayers benefit from adaptive reading and writing. We fix both read and write gates around either attention or MLP. Table 2 compares these restricted interfaces with full GR. MLP-only routing reaches 5.69 FID, outperforming attention-only routing (6.52) and approaching full GR (5.48). The two sublayer types therefore

Table 1: **Adaptive residual access.** Gates in blue are adaptive. GR readers retain independent branch normalization. Write weights multiply u; $\bar{a} = (a_1 + a_2)/2$.

Model	Residual computation	FID ↓
Baseline	h → Norm → r → Sublayer → u → h'	7.15
GR	h_1, h_2 → Read: g_1, g_2 → r → Sublayer → u → Write: a_1, a_2 → h'_1, h'_2	**5.48**
Read-only	h_1, h_2 → Read: g_1, g_2 → r → Sublayer → u → Write: $1, 1$ → h'_1, h'_2	5.97
Write-only	h_1, h_2 → Read: $1/2, 1/2$ → r → Sublayer → u → Write: a_1, a_2 → h'_1, h'_2	6.36
Shared-write	h_1, h_2 → Read: g_1, g_2 → r → Sublayer → u → Write: $\bar{a}, \bar{a}$ → h'_1, h'_2	6.09

benefit unequally from adaptive residual access, with the MLP-side interface providing the stronger restricted configuration.

Finding 2. *MLP-only adaptive routing outperforms attention-only routing; applying it to both sublayer types gives the best result.*

We retain adaptive reading and writing around both attention and MLP.[1]

4.2 From readout scale to direction calibration

Read scale. Adaptive gates change the content and the magnitude of the feature passed to the sublayer. For the GR reader in Equation 2, gates near the sigmoid's central value give $r_g \approx r_0/2$, where $r_0 = K^{-1} \sum_{i=1}^{K} z_i$ is the equal-weight mixture of normalized states. Since the mixed readout enters the sublayer without another normalization, this gate-induced scale change affects its input. We test whether this scale suits diffusion. We introduce a scalar multiplier on the mixed readout:

$$r_s = s\, r_g. \tag{4}$$

The original GR reader corresponds to $s = 1$. We compare it with $s = 2$, denoted Read×2, which centers the read near the ungated reference r_0. At unchanged parameter count and training budget, doubling the read scale improves FID from 5.48 to 5.01 (Table 3a). This experiment identifies readout scale as a consequential design choice for GR in diffusion. We use the resulting $r_2 = 2r_g$ as the uncalibrated reference for the following reader and writer comparisons.

Direction or magnitude? The scale result raises a further question: does adaptive reading help by selecting a direction, changing the magnitude, or both? To separate these effects, we construct a *direction-calibrated reader* that retains r_2's direction and matches r_0's RMS:

$$r_{\text{dir}} = r_2 \frac{\rho_\epsilon(r_0)}{\rho_\epsilon(r_2)}, \qquad \rho_\epsilon(v) = \sqrt{\max(D^{-1}\|v\|^2, \epsilon^2)}. \tag{5}$$

We use the ungated mean as the reference to restore the magnitude obtained from the same states without adaptive weighting, while preserving the direction selected by the gates. It remains input-dependent and varies with the normalized states, including their relative directions. Calibration therefore removes the gate-induced change in magnitude while retaining the scale of the current ungated read. Any common positive rescaling of r_2 cancels away from the numerical floor. A complementary choice, $r_{\text{mag}} = r_0\, \rho_\epsilon(r_2)/\rho_\epsilon(r_0)$, keeps the gate-induced magnitude but restores the reference direction. We compare direction and magnitude-only reading while keeping the original adaptive writer fixed (Table 3b). Direction reading achieves 4.98, compared with 5.69 for

[1] Inactive read and write gates use $g_i = 1/2$ and $a_i = 1$.

Table 2: **Adaptive routing location.** Read and write gates are adaptive at the indicated sublayers; the final reader remains adaptive.

Adaptive sublayers	Attention only	MLP only	Attention and MLP (GR)
FID ↓	6.52	5.69	**5.48**

Table 3: **Reader scale and geometry.** (a) Original GR ($s = 1$) versus Read×2 ($s = 2$). (b) Direction and magnitude controls. All variants retain the adaptive writer.

	(a) Read scale		**(b) Read geometry,** $s = 2$	
	$s = 1$	$s = 2$	Direction	Magnitude
FID ↓	5.48	**5.01**	**4.98**	5.69

magnitude-only reading. Preserving the direction selected by adaptive gates is therefore more effective than preserving their induced magnitude alone. We use direction calibration to retain adaptive feature selection at the ungated reference scale. This calibration adds no parameters.

Finding 3. *Doubling GR's read scale improves generation. Preserving adaptive direction is more effective than preserving adaptive magnitude alone.*

4.3 From update geometry to noise-conditioned writing

Which update component should be modulated? The writer updates the states available to subsequent blocks. We examine whether generation benefits more from modulating the entire update or selectively modulating its components parallel or orthogonal to the destination state. For the gated GR update $u_i = a_i u$, we decompose it relative to its destination state:

$$u_i^{\parallel} = h_i \frac{\langle h_i, u_i \rangle}{\|h_i\|^2}, \qquad u_i^{\perp} = u_i - u_i^{\parallel}. \tag{6}$$

The radial component acts along the pre-update state h_i; the tangent component is orthogonal to h_i.

Component and conditioning controls. We add a correction to the original GR update, $h_i' = h_i + u_i + \delta_\ell(t) v_i$, choosing v_i from the whole update u_i, the radial component $u_i^{\parallel}$, or the tangent component $u_i^{\perp}$. Since the same denoiser operates across noise levels, we let the correction strength depend on the timestep:

$$\delta_\ell(t) = \tfrac{1}{4} \tanh\big(w_\ell^\top \, \mathrm{RMSNorm}(e_t) + b_\ell\big). \tag{7}$$

Here $\mathrm{RMSNorm}(e_t)$ is gain-free RMS normalization of the timestep embedding. Each attention or MLP sublayer has a separate, zero-initialized controller. Its scalar output is shared across tokens and branches within a sample, while each branch uses its own projection. The three noise-conditioned controls have equal controller size and coefficient bounds. A static tangent control uses $\delta_\ell = \frac{1}{4}\tanh(b_\ell)$, with one zero-initialized scalar per sublayer.

Projection and noise conditioning. Table 4 shows that tangent writing reaches 4.72, compared with 4.95 for radial modulation and 5.02 for whole-update scaling. Tangent modulation achieves the lowest FID and preserves the original radial update. Replacing the timestep controller by a learned constant gives 4.93, so conditioning the correction on the noise level further improves FID.

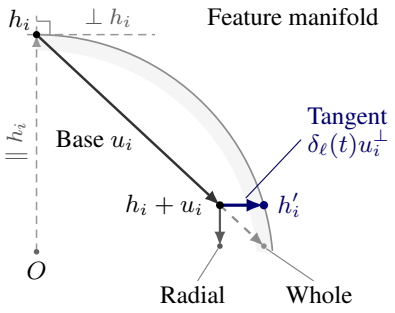
Finding 4. *The effective write control is state-relative and noise-dependent: modulating the tangent component outperforms both whole-update scaling and radial modulation.*

Noise-conditioned tangent writer. Our writer adds a noise-conditioned tangent correction to GR:

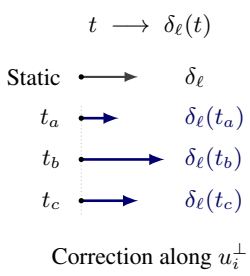
$$h_i' = h_i + u_i + \delta_\ell(t) u_i^{\perp}. \tag{8}$$

Table 4: **Writer geometry and noise conditioning.** (a) Corrections share the base point $h_i + u_i$, but $\|$ and $\perp$ refer to the pre-update state $h_i = \overrightarrow{Oh_i}$. Tangency is to the equal-norm sphere at h_i; proximity to the feature manifold illustrates our empirical intuition. (b) Timestep conditioning varies the correction strength; arrow lengths are schematic at fixed h_i, u_i. (c) Full adaptive $s = 2$ reader throughout; noise-conditioned controllers have equal size and coefficient bounds.

(a) Update geometry



(b) Noise conditioning



(c) Writer ablation

Correction v_i	Coeff.	FID ↓
None (0)	—	5.01
Whole u_i	$\delta_\ell(t)$	5.02
Radial $u_i^\parallel$	$\delta_\ell(t)$	4.95
Tangent $u_i^\perp$	δ_ℓ	4.93
Tangent $u_i^\perp$	$\delta_\ell(t)$	**4.72**

$$h_i' = h_i + u_i + \delta\, v_i$$

This writer leaves the radial component unchanged and varies the tangent coefficient within $(0.75, 1.25)$. Zero initialization recovers the original GR writer at the start of training. Post-training measurements show tangent suppression in earlier attention layers and amplification in later attention layers and most MLP layers, with noise-dependent coefficients in a subset of sublayers (Appendix E). The resulting change in update magnitude also depends on the state–update geometry.

Is tangent-only modulation sufficient? With the direction-calibrated reader fixed, we test two modifications to tangent writing. First, *update-norm calibration* rescales the tangent-modulated update $v_i = u_i + \delta_\ell(t)u_i^\perp$ toward the original update's norm, using $\widetilde{v}_i = v_i\sqrt{(\|u_i\|^2 + \epsilon)/(\|v_i\|^2 + \epsilon)}$. This scaling acts on both the radial and tangent components. Second, *independent radial/tangent modulation* allows both components to change, using $\widetilde{v}_i = u_i + \delta_t(t)u_i^\perp + \delta_r(t)u_i^\parallel$. The two coefficients use separately zero-initialized controllers of the form in Equation 7. Both variants write $h_i' = h_i + \widetilde{v}_i$.

Tangent-only modulation achieves 4.65 FID, compared with 5.19 for update-norm calibration and 4.94 for independent radial/tangent modulation. Neither modification improves generation, supporting our choice to modulate only the tangent component.

Finding 5. *Tangent-only modulation outperforms both update-norm calibration and independent radial/tangent modulation.*

4.4 Combining Reading and Writing

Table 5 compares the single-state interface and two GR readers with the tangent writer fixed.

Tangent writing across interfaces. With tangent writing, the single-state interface achieves 6.57 FID, while GR with an uncalibrated $s = 2$ read achieves 4.72. Direction calibration further improves FID to 4.65. Adaptive access, direction calibration, and tangent writing form the complete interface.

Finding 6. *With tangent writing, GR outperforms the single-state interface, and direction calibration gives the lowest FID among the tested interfaces.*

Does better writing remove the need for adaptive reading? We replace the adaptive $s = 2$ read by the equal-weight reference r_0, while retaining both residual streams, adaptive write gates, and the tangent controller. This raises FID from 4.72 to 5.87 at the same 200-epoch budget. Thus, storing two states and improving their updates does not recover the benefit of selecting their contributions to each sublayer. We retain adaptive reading when introducing the tangent writer.

Table 5: **Residual interfaces with tangent writing.** FID (↓) with the tangent writer fixed. The first row uses a single residual state; the two GR rows compare uncalibrated and direction-calibrated reads.

Residual interface	FID ↓
Single-state interface	6.57
GR, uncalibrated read ($s = 2$)	4.72
GR, direction-calibrated read	**4.65**

Finding 7. *Better writing does not replace adaptive reading. Even with a tangent writer, input-dependent access to the two states is important to generation quality.*

Complete interface. At each attention or MLP sublayer, we normalize the branches, form r_{dir}, evaluate $u = F_\ell(r_{\mathrm{dir}}; t, c)$ once, and write the gated output back using Equation 8. The final reader uses the same direction calibration before the prediction head. State width D and backbone attention and MLP dimensions remain unchanged. The controlled study establishes the effectiveness of GR in diffusion and identifies readout direction, scale, and noise-conditioned update geometry as consequential design choices. GR improves FID from 7.15 to 5.48; while the complete interface reaches 4.65. We next evaluate the interface across diffusion backbones.

5 Evaluation across diffusion backbones

5.1 Implementation details

Model implementation We evaluate our interface on class-conditional ImageNet generation with pixel-space JiT (Li & He, 2026) and PixelDiT (Yu et al., 2026), and latent-space SiT (Ma et al., 2024) and LightningDiT (Yao et al., 2025). We also test compatibility with representation alignment using SiT trained with REPA (Yu et al., 2025). + *Ours* denotes the direction-reader/tangent-writer interface. We use a state budget of $K = 2$ throughout the backbone evaluations. The width-D residual states supply a single input to each attention or MLP sublayer and receive branch-specific updates from its output. The same reader is used before the output head. Following GR, we set the reader bottleneck to $D/32$, giving $k = 24$ for JiT-B with $D = 768$ and scaling linearly with backbone width. The GR read and write gates retain their original initialization. The backbone's attention and MLP dimensions are retained, with each sublayer evaluated once per block. Direction calibration adds no trainable parameters; the tangent controllers add 18,456 parameters in JiT-B and are initialized to produce zero correction. Appendix D describes our optimized training and inference implementations, including fused-kernel pseudocode, an inference dataflow diagram, and performance measurements. We will publicly release the optimized kernels.

Evaluation protocol We evaluate ImageNet generation at 256×256 resolution using 50K class-conditioned samples and the ADM evaluation pipeline (Dhariwal & Nichol, 2021). We report generation FID (gFID) (Heusel et al., 2017), Inception Score (IS) (Salimans et al., 2016), precision, and recall (Kynkäänniemi et al., 2019). Conditional generation without classifier-free guidance (Ho & Salimans, 2022) uses CFG$= 1$; guided metrics use a CFG that follows the configuration from the baseline methods. We follow the original authors' evaluation protocols for each backbone. Model-specific configurations are provided in Appendix A.

5.2 Class-conditional image generation on ImageNet

We further evaluate whether the read/write design remains effective across generation spaces, training budgets, and representation alignment. Table 6 compares the interface with each backbone, following the corresponding training recipe.

Transfer across generation spaces. The same read/write interface improves guided FID across all backbone configurations in Table 6, spanning pixel-space JiT and PixelDiT and latent-space SiT and

Table 6: **Adding our read/write mechanism to existing methods on ImageNet-256.** Sampling configurations are aligned with their original settings. We report gFID, IS, precision (Prec.), and recall (Rec.) on 50K samples. "+ Ours" uses the direction-reader/tangent-writer interface. Better choice is in **bold**. Params reports inference-time denoiser parameters in millions.

			w/o CFG				w/ CFG			
Method	Params (M)	Epochs	gFID ↓	IS ↑	Prec.	Rec.	gFID ↓	IS ↑	Prec.	Rec.
Pixel-space Diffusion										
JiT-B/16	131.1	200	31.69	51.37	0.50	**0.65**	4.61	233.91	0.81	0.46
+ Ours	133.1	200	**28.69**	**55.85**	**0.52**	0.64	**4.32**	**249.62**	**0.82**	**0.46**
JiT-B/16	131.1	600	25.46	64.07	0.53	**0.66**	3.69	255.99	0.80	**0.52**
+ Ours	133.1	600	**22.86**	**69.72**	**0.56**	0.65	**3.55**	**272.35**	**0.81**	0.52
JiT-H/16	952.5	200	**9.20**	**123.93**	**0.67**	0.65	2.53	**291.06**	**0.81**	0.58
+ Ours	961.0	200	10.07	121.26	0.66	**0.65**	**2.46**	286.44	0.80	**0.59**
PixelDiT-XL/16	797.4	80	18.22	76.35	0.59	0.68	2.36	279.03	**0.79**	0.59
+ Ours	813.4	80	**16.59**	**79.70**	**0.60**	**0.69**	**2.08**	**282.58**	0.79	**0.61**
PixelDiT-XL/16	797.4	160	14.25	89.40	0.61	**0.69**	1.99	298.11	**0.80**	0.61
+ Ours	813.4	160	**14.14**	**90.99**	**0.61**	0.69	**1.96**	**298.48**	0.79	**0.62**
Latent-space Diffusion										
SiT-XL/2	675.1	80	18.46	72.23	0.64	0.63	5.70	162.30	0.80	**0.53**
+ Ours	681.9	80	**13.70**	**88.20**	**0.67**	**0.64**	**3.88**	**193.60**	**0.82**	0.53
LightningDiT-XL/1	675.3	64	5.14	130.20	**0.76**	0.62	2.11	252.30	**0.81**	0.58
+ Ours	681.9	64	**4.82**	**136.70**	0.75	**0.62**	**1.88**	**254.80**	0.80	**0.61**
Representation alignment										
SiT-XL/2 + REPA	675.1	80	7.74	**126.15**	**0.70**	0.65	1.91	**276.77**	**0.82**	0.60
+ Ours	681.9	80	**7.70**	124.72	0.70	**0.66**	**1.79**	273.99	0.81	**0.60**

LightningDiT. On SiT-XL/2, guided FID decreases from 5.70 to 3.88; the interface also improves the stronger LightningDiT reference from 2.11 to 1.88. JiT-B, PixelDiT, SiT, and LightningDiT additionally improve without guidance. These results show that the interface remains useful across different image representations and backbone designs.

Training budget and model scale. The benefit persists with longer training. JiT-B improves at both 200 and 600 epochs, reaching 3.55 guided FID at 600 epochs versus 3.69 for the baseline. PixelDiT also improves at both 80 and 160 epochs, although its margin narrows at the longer budget. The checkpoint comparison in Appendix B further traces JiT-B's gains from 60 to 600 epochs at a fixed CFG of 2.3. For the larger JiT-H, the improvement is specific to guided evaluation: FID decreases from 2.53 to 2.46.

Compatibility with representation alignment. The interface also improves SiT trained with REPA, which supervises intermediate features with an auxiliary representation objective. Adding our read/write design reduces guided FID from 1.91 to 1.79, while FID without guidance remains comparable (7.74 versus 7.70). Thus, the guided benefit extends to a backbone whose feature learning is already enhanced by representation alignment.

6 Conclusion

We presented a systematic study of how diffusion Transformers read from and write to multiple persistent residual states. Controlled comparisons demonstrate the value of adaptive access and identify readout scale, update geometry, and noise conditioning as consequential design choices. These findings motivate a direction-calibrated reader and a noise-conditioned tangent writer that retain shared sublayer computation. Their benefits extend across pixel- and latent-space backbones and training with representation alignment, establishing residual-state reading and writing as a complementary design axis for diffusion Transformers.

AI USE STATEMENT

We used generative AI tools to assist with code and method implementation, drafting and polishing the manuscript. The authors manually reviewed and verified all AI-assisted work at each stage and take full responsibility for the content of this paper, including its methods, results, and conclusions.

REPRODUCIBILITY STATEMENT

Sections 3 and 4 describe the residual-state interface and the controlled design variants. Implementation details and the evaluation protocol are provided in Section 5.1. Appendix A documents backbone-specific model, training, and sampling settings, while Appendix D provides pseudocode and profiling details for the optimized implementation. Code, including the optimized kernels, will be made publicly available.

REFERENCES

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. Rezero is all you need: fast convergence at large depth. In Cassio de Campos and Marloes H. Maathuis (eds.), *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 1352–1361. PMLR, 27–30 Jul 2021.

Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22669–22679, June 2023.

Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-σ: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 74–91, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-73411-3.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. 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. 12606–12633. PMLR, 21–27 Jul 2024.

Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Masked diffusion transformer is a strong image synthesizer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 23164–23173, October 2023.

Zipeng Guo, Lichen Ma, Yu He, Xiaolong Fu, Jingling Fu, Junshi Huang, and Yan Li. PixelU: A u-shaped transformer for efficient end-to-end pixel diffusion. In *Computer Vision – ECCV 2026*, volume 17060 of *Lecture Notes in Computer Science*, pp. 425–442, Cham, 2026. Springer. doi: 10.1007/978-3-032-37092-1_24.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. REPA-E: Unlocking VAE for end-to-end tuning of latent diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18262–18272, October 2025.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 36115–36125, June 2026.

Tianyu Li, Dongchen Han, Zixuan Cao, Haofeng Huang, Mengyu Zhou, Ming Chen, Erchao Zhao, Xiaoxi Jiang, Guanjun Jiang, and Gao Huang. Siamesenorm: Breaking the barrier to reconciling pre/post-norm. In *International Conference on Machine Learning*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Ilya Loshchilov, Cheng-Ping Hsieh, Simeng Sun, and Boris Ginsburg. ngpt: Normalized transformer with representation learning on the hypersphere. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 74014–74038, 2025.

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 23–40, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-72980-5.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, October 2023.

Qwen Team. On the design of Qwen3.8-Next architecture: Evaluation, efficiency, and training stability. Technical report, Alibaba Group, August 2026.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Jaeyo Shin, Jiwook Kim, and Hyunjung Shim. Representation alignment for just image transformers is not easier than you think. In *Computer Vision – ECCV 2026*, volume 17013 of *Lecture Notes in Computer Science*, pp. 603–620, Cham, 2026. Springer. doi: 10.1007/978-3-032-37271-0_33.

Jaskirat Singh, Xingjian Leng, Zongze Wu, Liang Zheng, Richard Zhang, Eli Shechtman, and Saining Xie. What matters for representation alignment: Global information or spatial structure? In *International Conference on Learning Representations*, pp. 33807–33845, 2026.

Kimi Team, Guangyu Chen, Yu Zhang, Jianlin Su, Weixin Xu, Siyuan Pan, Yaoyu Wang, Yucheng Wang, Guanduo Chen, Bohong Yin, Yutian Chen, Junjie Yan, Ming Wei, Y. Zhang, Fanqing Meng, Chao Hong, Xiaotong Xie, Shaowei Liu, Enzhe Lu, Yunpeng Tai, Yanru Chen, Xin Men, Haiqing Guo, Y. Charles, Haoyu Lu, Lin Sui, Jinguo Zhu, Zaida Zhou, Weiran He, Weixiao Huang, Xinran Xu, Yuzhi Wang, Guokun Lai, Yulun Du, Yuxin Wu, Zhilin Yang, and Xinyu Zhou. Attention residuals, 2026.

Hainuo Wang, Mingjia Li, and Xiaojie Guo. WiT: Waypoint diffusion transformers for alleviating trajectory conflict in pixel-space image generation. *arXiv preprint arXiv:2603.15132*, 2026.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. DeepNet: Scaling Transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(10):6761–6774, 2024. doi: 10.1109/TPAMI.2024.3386927.

Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, and Song Han. Sana: Efficient high-resolution text-to-image synthesis with linear diffusion transformers. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 20383–20407, 2025.

Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Mingyu Xu, Kuai Yu, Liang Zhao, Shangyan Zhou, Zhean Xu, Zhengyan Zhang, Wangding Zeng, Shengding Hu, Yuqing Wang, Jingyang Yuan, Lean Wang, and Wenfeng Liang. mhc: Manifold-constrained hyper-connections. In *International Conference on Machine Learning*, 2026.

Chao Xu, Maohua Li, Qirui Li, Yixuan Xu, Yanke Zhou, Yunhe Li, Cuifeng Shen, Hanlin Tang, Kan Liu, Tao Lan, Lin Qu, and Shao-Qun Zhang. Rethinking cross-layer information routing in diffusion transformer. In *Advances in Neural Information Processing Systems*, volume 39, 2026. In press.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15703–15712, June 2025.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 87400–87442, 2025.

Yongsheng Yu, Wei Xiong, Weili Nie, Yichen Sheng, Shiqiu Liu, and Jiebo Luo. PixelDiT: Pixel diffusion transformers for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14273–14282, June 2026.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 12360–12371, 2019.

Defa Zhu, Hongzhi Huang, Zihao Huang, Yutao Zeng, Yunyao Mao, Banggu Wu, Qiyang Min, and Xun Zhou. Hyper-connections. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 97183–97219, 2025.

A Model configurations and evaluation details

We list the model-specific settings for the backbone evaluations in Section 5.

JiT. We evaluate JiT-B/16 at 200 and 600 epochs and JiT-H/16 at 200 epochs. The global batch size is 1,024, with context start 4 and context length 32. Each reported epoch comprises 1,251 optimizer updates; the 200- and 600-epoch budgets therefore correspond to 250,200 and 750,600 updates. JiT-B/16 uses a patch-embedding bottleneck of 128, zero attention and projection dropout, and a denominator clamp of 0.05. Its AdamW optimizer (Loshchilov & Hutter, 2019) uses learning rate 2×10^{-4}, betas $(0.9, 0.95)$, zero weight decay, and five warmup epochs followed by a constant learning rate. JiT-H/16 has width 1280, depth 32, and 16 attention heads. Sampling uses Heun50 and EMA weights with decay 0.9999; The checkpoint comparisons in Table 7 use CFG=2.3.

PixelDiT. We evaluate PixelDiT-XL/16 (Yu et al., 2026) at 80 and 160 epochs on ImageNet-256×256, following the official training and inference settings. We apply our read/write interface to attention and MLP sublayers in both the patch and pixel stages, using $K = 2$. The reader bottleneck is fixed at $k = 64$, including the two stage-end readouts, rather than scaled with width. Per-stream normalization replaces the original pre-sublayer RMSNorm. Each stage maintains its own residual states. The patch-stage readout supplies the conditioning representation to the pixel stage, which initializes its states from the pixel embeddings.

SiT. SiT-XL/2 baseline and + Ours are trained for 400,000 optimizer updates at global batch 256, approximately 79.93 ImageNet epochs (reported as 80). Both use the linear interpolant and velocity-prediction objective, online SD-VAE ft-EMA encoding (Rombach et al., 2022) with fresh posterior samples and latent scale 0.18215, and AdamW with constant learning rate 10^{-4}, betas $(0.9, 0.999)$, and zero weight decay. Our interface uses $K = 2$ states. Sampling uses EMA weights with decay 0.9999. The released SDE Euler–Maruyama sampler uses 250 steps, diffusion coefficient $w_t = \sigma_t$, diffusion norm 1, and a final mean step of size 0.04. Guided evaluation uses CFG=1.5 over the full interval and guides the first three latent channels, following the upstream convention.

LightningDiT. We follow the official training and inference configurations of LightningDiT-XL/1 (Yao et al., 2025). Our model is trained for 80,000 optimizer updates at global batch 1,024, approximately 63.94 ImageNet epochs (reported as 64). It uses VA-VAE f16d32 and the released transport objective and optimizer recipe. Sampling uses EMA weights with decay 0.9999 and the official Euler sampler with 250 time points (249 model evaluations) and shift 0.3. Guided evaluation uses CFG=10 with the official guidance interval and channel convention.

B Additional evaluation results

B.1 Quality across training checkpoints

Table 7 follows JiT-B from 60 to 600 epochs at a fixed CFG of 2.3. This evaluation uses a common guidance scale across checkpoints, separate from the guided operating points in Table 6. The complete interface achieves lower FID than Baseline at every reported checkpoint.

Table 7: **JiT-B quality across training checkpoints (FID ↓).** Baseline versus the complete interface at fixed CFG 2.3, using the JiT-B configuration in Appendix A.

Epoch	60	80	100	120	140	160	180	200	220	240	260	280	300	320
Baseline	21.40	14.10	11.04	9.14	8.12	7.24	6.86	6.29	6.05	5.85	5.65	5.41	5.19	5.11
Ours	19.90	12.31	9.41	8.18	7.08	6.41	5.96	5.52	5.16	5.05	4.80	4.68	4.54	4.44
Epoch	340	360	380	400	420	440	460	480	500	520	540	560	580	600
Baseline	4.96	4.76	4.70	4.87	4.60	4.51	4.52	4.49	4.44	4.42	4.44	4.30	4.34	4.29
Ours	4.31	4.28	4.21	4.17	4.23	4.13	3.97	3.94	3.91	3.90	3.84	3.86	3.81	3.84

C Residual-state capacity

We examine generation quality and training cost as the number of residual states increases. Inference implementation and cost are reported in Appendix D.

C.1 Effect of the number of residual states

Increasing K expands persistent state while retaining one width-D attention or MLP computation. Table 8 compares $K \in \{2, 4, 6, 8\}$ for the direction-reader/tangent-writer interface on JiT-B/16. We include the original single-state backbone ($K = 1$) as a baseline. All models use 12 blocks of width 768, a patch-embedding bottleneck of 128, and 200 training epochs; our multi-state variants use a read-gate bottleneck of $k = 24$. Evaluation uses EMA weights (decay 0.9999); IS uses ten splits after shuffling samples. This is the aligned JiT-B configuration in Table 6; the main controlled

read/write ablation uses a patch-embedding bottleneck of 256. At the common guidance scale of CFG= 3.0, guided FID is 4.61 for the original baseline and improves from 4.32 with two states to 4.13 with four states; increasing to eight states yields a further reduction of only 0.03. Without CFG, six states achieve the lowest FID. More streams nevertheless increase the training cost even with our optimized read/write kernels, as quantified in the same table. The main study fixes $K = 2$ to compare read/write choices at a common state budget.

Table 8: **Generation quality and training cost versus residual-state count.** JiT-B/16 at 200 epochs. $K = 1$ is the original backbone; $K \geq 2$ uses our interface with optimized training kernels. Quality uses Heun50; guided results use CFG= 3.0, matching the 200-epoch JiT-B rows of Table 6. Training cost uses the common single-GPU, batch-32 protocol below; relative step time is normalized to $K = 1$. Underlines mark the best quality metrics.

	w/o CFG	w/ CFG (= 3.0)				Training cost			
K	FID ↓	FID ↓	IS ↑	Prec. ↑	Rec. ↑	Step (ms) ↓	Relative	img/s ↑	Peak (GiB) ↓
1 (baseline)	31.69	4.61	233.91	0.81	0.46	42.42	1.00×	754.03	7.65
2	28.69	4.32	249.62	<u>0.82</u>	0.46	59.40	1.40×	537.76	10.15
4	28.60	4.13	<u>258.05</u>	0.82	<u>0.48</u>	101.48	2.39×	312.93	13.26
6	<u>27.76</u>	4.19	255.92	0.82	0.47	129.61	3.06×	246.84	16.14
8	27.95	<u>4.11</u>	256.68	0.82	0.47	156.40	3.69×	204.54	18.97

Training cost with optimized kernels. All models in Table 8 use the same execution settings. The fused CuTe/Triton two-stream pipeline for $K = 2, 4, 6, 8$. These measurements use the training implementations, including their backward kernels. The original backbone takes 42.42 ms per step and 7.65 GiB peak memory; the optimized two-stream interface takes 59.40 ms (+40.0%) and 10.15 GiB. Despite these optimizations, increasing from 2 to 8 streams raises step time by 163.3% and peak allocated memory by 86.9%. Going from four to eight streams costs 54.1% more step time for a guided FID reduction of only 0.03. Sharing the attention and MLP computation does not remove the state-dependent normalization, read/write, and backward work.

Measurement protocol. We profile JiT-B/16 at 256×256 on one RTX PRO 6000 Blackwell Server Edition (96 GB), with batch 32 and common execution settings for every K. Each step includes forward and backward computation, AdamW, and an EMA update. We report median step time and throughput over repeated measurements after warmup, together with peak allocated memory. Measurements use initialized model weights and synthetic inputs on the GPU, excluding data loading, distributed communication, and evaluation.

D Efficient Implementation

We implement optimized CuTe/Triton kernels for both training and inference. The training paths support $K \in \{2, 4, 6, 8\}$ and include backward kernels; Table 8 reports their training cost. We will release the implementation and benchmark code.

Read/write fusion. Although attention and MLP computation is shared, each sublayer still normalizes, reads, and updates multiple residual states. Implementing these operations separately creates intermediate tensors and repeated memory transfers. We fuse the current write with the next reader's normalization and down-projection, then combine up-projection, direction calibration, and backbone conditioning in a second kernel (Figure 3). Buffers are reused after their last consumer. Context insertion and the final readout use separate boundary paths. We validate the fused operations against the unfused reference implementation.

Fused operations. Algorithms 1–3 expand the operations inside each fusion. We show one token with two width-D states, $H = (h_1, h_2)$, and reader rank k; the measured setting uses $D = 768$ and $k = 24$. The next sublayer's parameters carry a superscript $+$. v denotes the current sublayer's output before the backbone output gate; u includes that gate, as in the main text.

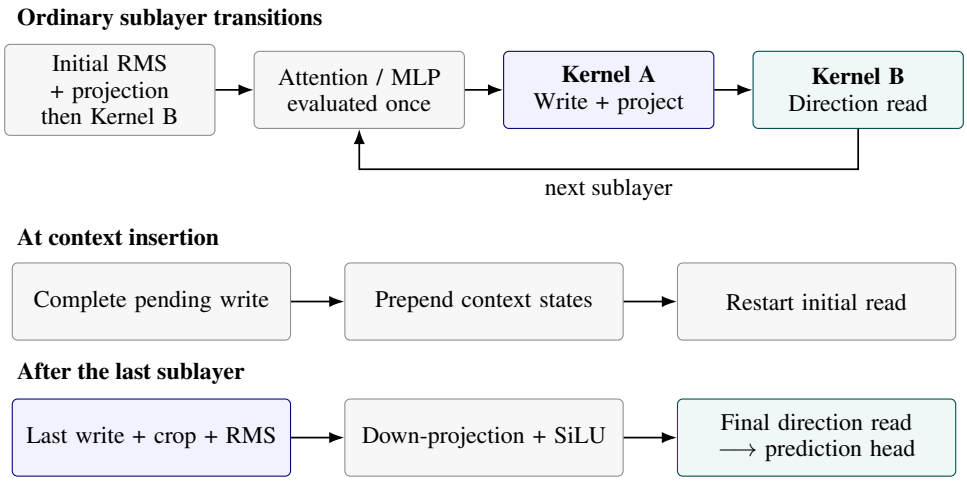


Figure 3: **Inference dataflow.** The ordinary path alternates backbone sublayers with two fused kernels (Algorithms 1 and 2). Context insertion restarts the read; Algorithm 3 describes the final path over image tokens.

Algorithm 1 Kernel A: fuse the current write with the next normalization and projection

Input: States H; sublayer output v; output gate e; write logits q; $\delta(t)$
Input: Next reader's γ^+, W^+_{down}, and W^+_{inj}
Output: Updated H; inverse RMS ρ; rank-k feature b; next write logits q^+

$u \leftarrow v \odot e$ ▷ Apply backbone output gate
for $i = 1, 2$ **do**
  $a_i \leftarrow 2\,\text{sigmoid}(q_i)$
  $d_i \leftarrow a_i u$ ▷ GR injection
  $\lambda_i \leftarrow \langle h_i, d_i \rangle / \max(\|h_i\|^2, \epsilon^2)$ ▷ Radial coefficient
  $h_i \leftarrow h_i + d_i + \delta(t)(d_i - \lambda_i h_i)$ ▷ Tangent correction
  $\rho_i \leftarrow \left(D^{-1}\|h_i\|^2 + \epsilon\right)^{-1/2}$
  $x_i \leftarrow (h_i \rho_i) \odot \gamma_i^+$ ▷ Keep in shared memory
end for
$W \leftarrow [W^+_{\text{down}}; W^+_{\text{inj}}], \quad x \leftarrow [x_1; x_2]$
$[p; q^+] \leftarrow \frac{1}{2} W x$ ▷ One joint projection
$b \leftarrow \text{SiLU}(p)$
Store H, ρ, b, q^+; discard shared x

Algorithm 2 Kernel B: fuse up-projection, direction reading, and input modulation

Input: H, ρ, b from Kernel A; next reader's $\gamma^+, W^+_{\text{up}}$; modulation scale α, shift β
Output: Modulated input r for the next attention or MLP sublayer

$q_{\text{read}} \leftarrow W^+_{\text{up}} b$ ▷ Logits stay in shared memory
for $i = 1, 2$ **do**
  $n_i \leftarrow (h_i \rho_i) \odot \gamma_i^+$ ▷ Reconstruct normalized states
  $g_i \leftarrow 2\,\text{sigmoid}(q_{\text{read},i})$
end for
$m \leftarrow \frac{1}{2}(g_1 \odot n_1 + g_2 \odot n_2)$ ▷ Adaptive mixture
$r_0 \leftarrow \frac{1}{2}(n_1 + n_2)$ ▷ Ungated reference
$A_0 \leftarrow \sqrt{\max(D^{-1}\|r_0\|^2, \epsilon^2)}, \quad A_m \leftarrow \sqrt{\max(D^{-1}\|m\|^2, \epsilon^2)}$
$\kappa \leftarrow A_0 / \max(A_m, \epsilon)$ ▷ Match the reference magnitude
$r \leftarrow (m\kappa) \odot (1 + \alpha) + \beta$
Store r in the consumed v buffer; discard shared q_{read}

Algorithm 3 Terminal path: fuse image-token writing and normalization, then final reading

Input: Last states and update (H, v, e, q, δ); final reader $(\gamma, W_{\text{down}}, W_{\text{up}})$
Output: Feature for the prediction head, containing image tokens only

Fused write–crop–RMS kernel, over image tokens only:

Compute temporary $\tilde{h}_i, \rho_i$ using Kernel A's write and normalization; keep H unchanged
Store $Y_i \leftarrow \tilde{h}_i$, ρ_i, and $x_i \leftarrow (\tilde{h}_i \rho_i) \odot \gamma_i$ in new buffers

Separate down-projection and activation stage:

$p \leftarrow \frac{1}{2} W_{\text{down}}[x_1; x_2]$
$b \leftarrow \text{SiLU}(p)$

Fused final up-projection and direction-read kernel:

Compute $q_{\text{read}}, n_i, g_i, m, r_0, \kappa$ as in Kernel B, from Y, ρ, b
Store $r \leftarrow m\kappa$ ▷ No backbone input modulation

Storage and boundaries. Kernel A updates the internal state buffer; Kernel B reuses the sublayer-output buffer only after Kernel A has consumed it. At context insertion, we complete the pending write before prepending context states and restarting the reader (Figure 3). Unlike the ordinary path, the terminal path stores a cropped state and a normalized projection input. Local allocation savings therefore do not imply a decrease in whole-model peak memory.

Inference cost. Table 9 separates the cost of the read/write design from the benefit of kernel fusion. With standard PyTorch operations, our interface adds 3.40% forward latency over GR. Fusion increases throughput over our naive implementation by 33.92%, reaching 87.37% of backbone throughput. Peak memory rises from 36.69 to 38.41 GiB.

We profile one denoiser forward at $t = 0.5$ using initialized model weights and synthetic inputs on the GPU, without CFG or a sampler. All methods use the same hardware and execution settings. After warmup, we report median throughput and batch latency over six repeats, and peak allocated memory across two runs. Data loading and compilation are excluded.

Table 9: **Single-forward inference cost.** JiT-B/16 at 256×256, batch 4096, on one RTX PRO 6000 Blackwell Server Edition. The backbone uses $K = 1$; GR and Ours use $K = 2$ and reader rank 24. Naive uses standard PyTorch operations; fused uses our custom kernels. All methods share the same execution settings.

Implementation	img/s ↑	Forward (ms) ↓	Peak (GiB) ↓
Original backbone	2563.38	1597.89	33.30
GR (naive)	1729.32	2368.57	36.69
Ours (naive)	1672.40	2449.17	36.69
Ours (fused)	2239.74	1828.79	38.41

E LAYER-WISE BEHAVIOR OF THE TRAINED TANGENT WRITER

Measurement setup. We characterize how the trained writer modulates updates across depth and noise levels. We analyze the controlled-study JiT-B/16 checkpoint of Ours at 200 epochs (250,200 optimizer steps), using EMA weights with decay 0.9999. For 256 ImageNet validation images and two fixed Gaussian noise draws per image, we evaluate noise fractions $\sigma \in \{0.05, 0.50, 0.90\}$, where $x_\sigma = (1-\sigma)x + \sigma\epsilon$ and the model timestep is $t = 1 - \sigma$. We record all 12 attention and 12 MLP writers without modifying their outputs. Update statistics use spatial tokens and average over tokens, branches, images, and noise draws; block indices below start at zero.

Layer-specific modulation. The learned coefficients exhibit a consistent pattern across depth at the three measured noise levels. Writing the branch update before tangent correction as $u_i = u_i^{\parallel} + u_i^{\perp}$, the corrected update is

$$\widetilde{u}_i = u_i^{\parallel} + \big(1 + \delta_\ell(t)\big)u_i^{\perp}. \tag{9}$$

Thus, a negative coefficient suppresses the tangent component, while a positive coefficient amplifies it, relative to the same model's uncorrected update. Table 10 shows negative coefficients at attention blocks 0–8 and positive coefficients at blocks 9–11 at all three noise levels. Among MLP writers, 10 of 12 coefficients are positive at $\sigma = 0.05$, and 11 of 12 are positive at $\sigma = 0.50$ and 0.90. Many coefficients lie near the controller bounds of -0.25 and $+0.25$. Overall, 23 of the 24 sublayers retain their coefficient sign across the measured noise levels. The exception is MLP block 7, which switches from mild suppression at low noise to amplification at higher noise.

Table 10: **Learned tangent coefficients in JiT-B/16.** Coefficients $\delta_\ell(1-\sigma)$ for the 200-epoch EMA checkpoint of Ours. Positive and negative values amplify and suppress the tangent component, respectively; the radial component is unchanged. Values are rounded to three decimals.

	Attention			MLP		
Block	$\sigma = .05$	$\sigma = .50$	$\sigma = .90$	$\sigma = .05$	$\sigma = .50$	$\sigma = .90$
0	-0.241	-0.249	-0.250	$+0.250$	$+0.250$	$+0.250$
1	-0.250	-0.250	-0.250	$+0.250$	$+0.250$	$+0.250$
2	-0.250	-0.250	-0.250	$+0.250$	$+0.250$	$+0.250$
3	-0.250	-0.250	-0.250	$+0.250$	$+0.250$	$+0.250$
4	-0.209	-0.236	-0.250	$+0.250$	$+0.250$	$+0.250$
5	-0.122	-0.217	-0.250	-0.250	-0.250	-0.250
6	-0.250	-0.250	-0.250	$+0.178$	$+0.237$	$+0.250$
7	-0.250	-0.250	-0.250	-0.055	$+0.225$	$+0.250$
8	-0.250	-0.250	-0.250	$+0.203$	$+0.245$	$+0.250$
9	$+0.250$	$+0.250$	$+0.250$	$+0.250$	$+0.250$	$+0.250$
10	$+0.250$	$+0.250$	$+0.250$	$+0.250$	$+0.250$	$+0.250$
11	$+0.250$	$+0.250$	$+0.250$	$+0.250$	$+0.250$	$+0.250$

Effect on actual updates. The coefficient pattern is accompanied by changes in update magnitude, while the resulting magnitude ratio also depends on the state–update geometry. At $\sigma = 0.50$, the mean ratios $\|\widetilde{u}_i\|/\|u_i\|$ are 0.862 and 0.753 for attention blocks 1 and 7, whose coefficients are both approximately -0.25. In contrast, the ratios are 1.182 for attention block 10 and 1.220 for MLP block 3, whose coefficients are approximately $+0.25$. A nearly constant coefficient does not imply a constant effect across noise levels: attention block 1 retains $\delta \approx -0.25$, but its mean magnitude ratio increases from 0.804 to 0.862 to 0.908 as σ increases from 0.05 to 0.50 to 0.90. These measurements show layer-specific tangent suppression and amplification, with noise-dependent changes in a subset of sublayers.