Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Analog Diffusion Models

Jiaqi Chu

Jiaqi.Chu@microsoft.com

Microsoft Research https://orcid.org/0009-0008-4744-334X

Heiner Kremer

Microsoft Research

Fabian Falck

Microsoft Research

Grace Brennan

Microsoft research https://orcid.org/0000-0001-7081-0509

Burcu Canakci

Microsoft Research

James Clegg

Microsoft Research Ltd https://orcid.org/0009-0002-3428-7161

Daniel Cletheroe

Microsoft Research (United Kingdom) https://orcid.org/0009-0003-6444-9149

Doug Kelly

Microsoft Research

Christos Gkantsidis

Microsoft Research https://orcid.org/0000-0002-6898-2368

Michael Hansen

Microsoft Research

Paul Jeha

Microsoft Research

Kirill Kalinin

Microsoft Research

Jim Kleewein

Microsoft

Babak Rahmani

Microsoft Research

Saravan Rajmohan

Microsoft

Victor Rühle

Microsoft https://orcid.org/0000-0002-8957-7628

Jannes Gladrow
Microsoft Research
Francesca Parmigiani
Francesca.Parmigiani@microsoft.com
Microsoft Research https://orcid.org/0000-0001-7784-2829
Hitesh Ballani
Microsoft Research https://orcid.org/0000-0003-1573-3314

Physical Sciences - Article

Keywords: generative AI, analog computing, diffusion models, flow matching, optical computing, neuromorphic computing, fixed-point compute paradigm

Posted Date: August 6th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8919479/v2

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Analog Diffusion Models

Heiner Kremer[*,1] Fabian Falck[*,1] Jiaqi Chu[*,1,†] Grace Brennan[1] Burcu Canakci[1] James H. Clegg[1] Daniel Cletheroe[1] Douglas J. Kelly[1] Christos Gkantsidis[1] Michael Hansen[2] Paul Jeha[1] Kirill P. Kalinin[1] Jim Kleewein[2] Babak Rahmani[1] Saravan Rajmohani[2] Victor Rühle[2] Jannes Gladrow[1] Francesca Parmigiani[1,‡] & Hitesh Ballani[1]

[1]Microsoft Research, Cambridge, UK,

[2]Microsoft, Redmond, WA, USA

[†]Correspondence to: Jiaqi.Chu@microsoft.com

[‡]Correspondence to: Francesca.Parmigiani@microsoft.com

Analog and optical computing hardware offers orders-of-magnitude efficiency gains for AI inference, but exploiting those gains requires running AI workloads on such hardware while remaining compatible with the model ecosystem. This combination has proved elusive. Diffusion models, key workloads in generative AI, present this challenge acutely: their iterative inference is compute- and energy-intensive, yet prior analog implementations either modify the model formulation or remain partially digital, limiting compatibility or efficiency. Our key observation is a fixed-point correspondence between the physical equilibration dynamics of analog hardware and implicit diffusion inference. We introduce Analog Diffusion Models (ADMs) that exploit this correspondence to execute diffusion inference fully on analog hardware while keeping the model and its training conventional and fixed-point-free. On an iterative analog optical computer with 2,304 programmable weights, we demonstrate fully analog generation of two-dimensional and latent-space distributions, with measured convergence times of 10–15 µs per step. Digital-twin simulations across convolutional and transformer architectures (32M–13B parameters) show that the algorithm scales: ADMs match standard generation quality with up to 16× fewer diffusion steps, sufficient to offset the additional fixed-point iterations per step, preserving system-level efficiency gains. These results raise the possibility that exploiting structural alignment between hardware dynamics and algorithm requirements, rather than algorithm redesign, may offer a broader and practical route to realizing analog computing's efficiency potential across AI workloads.

Introduction

Generative artificial intelligence (GenAI) has emerged as one of the most transformative technological developments of recent years, with diffusion models (*1, 2*) becoming the dominant paradigm for generating images (*3*), audio (*4*), video (*5, 6*), and scientific data (*7, 8*). Their success comes at a cost: each sample requires many sequential forward passes to transform noise into data, making inference compute-, energy-, and latency-intensive. As these models scale, inference accounts for an estimated 90% of energy use in commercial deployments (*9*).

Analog and optical hardware seems the natural platform for diffusion inference as it can offer orders-of-magnitude reductions in latency and energy consumption for the compute-intensive matrix multiplications central to AI workloads (*10–27*). Yet early demonstrations (*10, 18, 20, 21*) expose a persistent gap between hardware-level and application-level gains. Some confine analog hardware to a small component within a larger digital pipeline, preserving standard model and training but limiting system-level benefit (*18*). Others redesign the generative model around hardware constraints (*10, 18, 20, 21*), sacrificing compatibility with standard training pipelines, pretrained weights, and the rapidly improving algorithms of the mainstream ecosystem. In either case, hardware-level efficiency does not reach the user.

This gap is reinforced by a structural lock-in. The community develops and benchmarks diffusion advances on digital accelerators, so algorithms that map well to dominant platforms are the ones optimized and adopted: the so-called hardware lottery (*28*). Alternative hardware that cannot directly leverage these improvements risks irrelevance regardless of its raw efficiency. For unconventional computing to deliver on its promise, it must therefore offer application-level benefits at inference and remain compatible with the standard diffusion models and training pipelines that the broader AI community continuously improves. To date, no system has demonstrated both.

Here, we observe that each implicit diffusion step has the same convergence structure as the physical equilibration of an analog feedback loop, allowing the analog system to execute

the diffusion update natively. Building on this correspondence, we introduce Analog Diffusion Models (ADMs) and demonstrate the first fully analog execution of diffusion inference while keeping the model and its training conventional and compatible with the standard ecosystem.

While ADM inference may be implemented on a range of analog platforms with suitable equilibration dynamics (*29–35*), here we demonstrate it on an Analog Optical Computer (AOC) with 2,304 programmable weights, scaled tenfold from our earlier fixed-point computing platform (*27*). The AOC performs fixed-point search entirely in the analog domain by combining three-dimensional optics and analog electronics in a feedback loop until convergence (Fig. 1A). The fixed-point view relates to prior deep-equilibrium approaches in the digital domain (*36–39*); ADMs differ by requiring no fixed-point search at training, as implicit and standard sampling share the same trained denoiser, and by mapping inference onto physical analog hardware. Training remains digital: ADMs are optimized using conventional diffusion-model pipelines on GPUs. To account for analog-system non-idealities, we train models with a digital twin (DT) of the hardware. The DT acts as a task-agnostic, differentiable translation layer that applies hardware-specific alterations and corruptions to model parameters and outputs while leaving the neural network itself in standard form. Unlike approaches that incorporate hardware-in-the-loop training or iterative digital-twin refinement (*18, 40–42*), we employ a single DT whose non-model parameters remain fixed across all experiments, eliminating the need for hardware evaluations during training and allowing the approach to extend to a wide range of hardware implementations, including those unsuitable for backpropagation-based training (*40*).

We benchmark ADMs in three settings spanning analog execution, image generation, and algorithmic scalability. First, we demonstrate fully analog diffusion steps on hardware using diverse two-dimensional distributions (Fig. 1B, Fig. 2A), with measured convergence times of 10–15 μs per diffusion step. Second, using a latent diffusion framework (*43*), we execute the iterative diffusion process entirely on the hardware, performing only the final decoding step digitally, to generate MNIST, FashionMNIST, and ExtendedMNIST samples (Fig. 1C). Third, to

demonstrate the algorithmic scalability of implicit diffusion sampling beyond the current hardware prototype, we conduct digital-twin simulations using common backbone architectures, including convolutional U-Nets (*44*) and diffusion transformers (DiT) (*45*), on high-resolution image datasets such as Smithsonian Butterflies, AFHQ, and Oxford Flowers-102. Across architectures, datasets, and model sizes ranging from 32 million to 13 billion parameters, ADMs consistently reduce the number of diffusion steps required to reach a given sample quality, achieving the same Fréchet Inception Distance (FID) as conventional methods, with up to 16× fewer steps. Although each implicit step requires multiple fixed-point iterations rather than the single function evaluation of standard explicit sampling, the trade-off breaks even in total computation and can reduce total network evaluations by 1–2×. This means that the additional fixed-point iterations do not offset the benefit of using fewer diffusion steps, allowing analog per-operation efficiency gains to carry through the denoising workload. Combined with projected hardware energy gains (Fig. 1E) from next-generation miniaturized implementations, these results suggest that matching hardware physics to algorithmic structure, rather than adapting algorithms to hardware, may be a more general route to unlocking analog computing's efficiency for AI workloads.

Analog realization of fixed-point diffusion steps

Diffusion models iteratively transform Gaussian noise into samples from a target distribution by integrating either a stochastic or an ordinary differential equation (ODE) (*46, 47*). Here we focus on ODE-based diffusion variants, specifically the recently proposed flow-matching formulation (*47–49*). Our method extends directly to other deterministic sampling methods like Denoising Diffusion Implicit Models (*50*).

While the iterative nature of ODE solvers is conceptually compatible with analog hardware, direct implementation faces two mismatches. First, diffusion models integrate their underlying ODE over a finite time horizon, typically from $t = 0$ to $t = 1$, whereas analog systems evolve according to physical relaxation dynamics toward a stationary state, effectively solving the ODE as $t \to \infty$. Second, early termination at a precise time is generally difficult on analog systems: unlike digital implementations with well-defined discrete iterations, many analog platforms (*20, 21, 27*) evolve through continuous-time dynamics without an intrinsic notion of iteration count. To address these issues, we implement the diffusion process using an implicit ODE solver, where each step corresponds to a fixed-point search. The choice of ODE solver affects the number of steps required to reach a given sample quality (*51, 52*). While conventional diffusion solvers employ explicit discretizations of the underlying ODE (Equation 3), ADMs rely on an implicit discretization scheme (Equation 4), requiring the solution of a nonlinear system at each step. This formulation naturally aligns with equilibrating analog hardware, defining each diffusion step as the stable fixed-point of a learned nonlinear transformation conditioned on the previous state and diffusion time.

This formulation avoids the stability limitations of explicit discretizations in rapidly varying vector field regimes near the data manifold and pure noise (*46*). It further provides robustness to analog noise, arising from two complementary mechanisms. First, the attractor dynamics of the fixed-point iteration make each diffusion step inherently robust to noise (fig. S13B left). Second, the diffusion process itself further suppresses noise, since each denoising step is designed

to operate on noisy inputs, with additional analog noise effectively absorbed into the diffusion noise (fig. S13B right; C).

Model parameters are learned using standard diffusion objectives on digital GPUs, without fixed-point search during training. Because analog non-idealities affect inference-time behavior, in analog models training and inference are inherently coupled: to reflect the non-idealities correctly, inference needs to be based on exactly the same operations as training, which in this case would require prohibitively costly fixed-point training. We avoid this by capturing these non-idealities in the DT and exploiting a duality between explicit and implicit Euler discretization schemes. This ensures the competing goal of consistency between training and inference while preserving standard training procedures, yet enables implicit ODE discretizations in ADMs that are naturally suited to equilibrating analog hardware with inherent noise. Details on methodology and implementation are provided in supplementary materials.

We demonstrate ADMs on AOC that integrates 3D optical and analog electronic technologies in a recursive loop. We use micro light-emitting diode (microLED) arrays, reflective spatial light modulators (SLMs) with pixels encoding model weights, and photodetector (PD) arrays, leveraging fan-out and fan-in of light in the third dimension to enable inherently parallel, scalable matrix multiplications. Compared to the system presented in (*27*), our hardware is scaled up by a factor of about 10×, to 2,304 programmable optical weights, and has been re-designed (e.g. implementing two independent optical paths instead of a single pass to represent positive and negative weights simultaneously) to reduce noise, mitigate non-idealities, and minimize discrepancies between experimental results and DT (refer to fig.S1 and supplementary materials A for additional hardware details). Fig. 1A describes the modular design for implementing multi-layer diffusion models on non-traditional hardware: each layer of the neural network is represented by a module, implementing optical matrix-multiplication and analog non-linearity. Multiple modules are connected sequentially through a crossbar analog switch, and the final module is connected back to the first to form a closed loop for the implicit scheme. The same

principle can be implemented using a single module with one of the blockmatrix structures exemplarily shown in the inset. The latter approach has been used in our experiments. Representative 2D distributions generated on hardware are shown on the right. Fig. 1B visualizes the diffusion process using hardware results. Each diffusion step consists of a fixed-point search (columns). Once converged, the result of the diffusion step is used to condition the next fixed-point search (rows). On this system, fixed-point inference for 2D distributions takes 10–15 μs per diffusion step, governed by the current system bandwidth of 5 MHz (-3dB; fig. S9B). Such per-step latencies are expected to be on the order of tens of nanoseconds in future hardware generations with target modulation speeds of a few GHz and centimeter-scale hardware footprints (*11, 53*).

Fig. 2A presents samples from a diverse set of 2D distributions generated on the same hardware, showing that analog fixed-point search reliably implements diffusion steps despite hardware noise and device nonidealities. Across the 2D target distributions, the hardware results closely match the DT results, with a per-step normalized root mean square error (NRMSE) around 10^{-2} (fig. S10C). For quantitative benchmarking, we compare distributional discrepancies of the generated distributions to the target distribution with those of a Gaussian distribution used as a metric reference (Fig. 2B and C; *Supplementary Section H* for metric definitions). In all cases, our results significantly improve over the Gaussian reference. To reduce remaining stochastic errors, we average results either across multiple experimental runs (fig. S7A) or over convergence windows (fig. S7B), which yields stable outputs. Due to the noise robustness, five repetitions corresponding to convergence window of 0.2 μs is sufficient.

Latent ADMs on hardware

Following the state-of-the-art in diffusion modeling, we employ a latent diffusion framework (*43*), performing the diffusion process in the latent space of a variational autoencoder (VAE). At training time, an ADM learns the latent space distribution of the VAE-encoded data. At

inference time, the latent ADM is evaluated on the hardware and the final result mapped to the data space using the VAE decoder (Fig. 3A). The diffusion process runs entirely in the analog domain, while only the final decoding step remains digital. In latent diffusion at scale, energy consumption is dominated by the iterative diffusion process in latent space, whereas VAE decoding is a single one-time operation, making the diffusion process the key bottleneck and particularly well suited for analog hardware acceleration. The all-analog nature of our diffusion process is in contrast to prior optical diffusion work (*19*), where a large digital decoder network is employed in every diffusion step, leaving only a small fraction of the diffusion process to be implemented in the optical domain.

Using the latent ADM approach we generate samples from the canonical MNIST, FMNIST and EMNIST datasets. Fig. 3B compares samples generated with ADM on the AOC hardware with those obtained using ADM on the DT and a standalone VAE (VAE-alone). Results show that ADMs (hardware) substantially outperform the VAE-alone baselines qualitatively (Fig. 3B) and quantitatively, yielding lower FID in the data space (Fig. 3C) and lower distributional discrepancies in the latent space (Fig. 3D). Figure S8 provides additional representative results for EMNIST, showing the evolution of the ADM (hardware) samples across diffusion steps and the robustness to analog noise across trials and time intervals. Figure S14 further shows close correspondence between the latent distributions generated by the hardware, the DT and the ground truth on MNIST. These benchmarks demonstrate that latent ADMs provide a hardware-compatible pathway for high-dimensional generative modeling with our prototype hardware, enabling high-fidelity image synthesis while confining all iterative diffusion computations to the analog domain.

ADMs at scale on the digital twin

We next evaluate algorithmic scaling beyond the present AOC hardware. In DT simulations, we test the implicit sampling scheme of ADMs on multiple high-dimensional image datasets

comprising Smithsonian Butterflies, AFHQ, and Oxford Flowers-102, and across a range of model architectures. These include convolutional U-Nets, DiT, and a scaled-up DT of our analog hardware without non-idealities. In addition to the implicit Euler scheme discussed so far (labeled as Implicit), we investigate the Crank-Nicolson solver (Implicit-CN), an implicit variant that combines the implicit update with an additional explicit model evaluation, enabling direct compatibility with our hardware (see supplementary materials). We compare these approaches with the explicit Euler scheme (Explicit) employed in state-of-the-art models (*6, 54*). We consider 256×256 image generation with model sizes ranging from 32 million to 13 billion parameters, training in the Stable Diffusion VAE latent space (*6*) and decoding generated latents with TAESD for the 32–100M models or with the Stable Diffusion VAE decoder for the 13B setting. All our models are trained using the standard flow matching objective without loss weighting or rectification steps. For each experiment the same model is evaluated with different schedulers. Architectural details and training hyperparameters are provided in Supplementary Section F.3.

Across datasets and backbone architectures, implicit sampling consistently outperforms explicit sampling in the low-step regime, while converging to the same asymptotic sample quality as the number of diffusion steps increases (Fig. 4A). For example, on Oxford Flowers-102 using a U-Net backbone, the implicit sampler, and similarly Implicit-CN, achieves an FID of 26.1 using only 8 diffusion steps (with 4 fixed point iterations per step), outperforming the explicit variant with 128 steps (FID 30.2), corresponding to a 16× reduction in steps and 4× reduction in total neural network evaluations (algorithmic improvement), see middle graph in Fig. 4A. On Smithsonian Butterflies, both implicit variants reduce diffusion steps by 4× compared with explicit sampling, corresponding to up to a 2× reduction in total network evaluations, see left most graph in Fig. 4A. On AFHQ with DiT, the implicit sampler shows no advantage. This is explained by the stability of the fixed-point search. Stable convergence imposes a lower bound on the number of diffusion steps, $T_{\min}$, which scales with the Lipschitz constant of the

learned vector field v, $T_{\min} \sim \mathrm{Lip}(v)$, see Theorem 1 in supplementary materials. In the case of AFHQ with DiT, this constraint limits the effectiveness of basic implicit methods. Here, the Crank–Nicolson variant, Implicit-CN, provides a more favorable trade-off. As a second-order method with $\mathcal{O}(\Delta t^2)$ global error, compared to $\mathcal{O}(\Delta t)$ for explicit and implicit Euler, it improves accuracy at larger step sizes, while its fixed-point formulation admits a more relaxed stability condition, with $T_{\min,\mathrm{CN}} \sim 2 \times \mathrm{Lip}(v)$. On AFHQ, where the Implicit scheme is unstable at low diffusion step counts, Implicit-CN restores stable fixed-point convergence and recovers the efficiency gains of implicit sampling. This yields an up to 8× reduction in diffusion steps, comparing 8 steps of Implicit-CN (FID 19.3) with 64 steps of Explicit (FID 19.6) and 1-2× reduction in total neural network evaluations, see most right graph of Fig. 4A.

Fig. 4B shows qualitative samples comparing implicit variants (Implicit for Smithsonian Butterflies and Oxford Flowers-102, and Implicit-CN for AFHQ) with the explicit Euler scheme at low diffusion step counts across datasets and architectures. For the same number of steps, implicit sampling yields sharper images with clearer structure and greater sample diversity, whereas the explicit baseline exhibits noticeable blur and loss of detail. These qualitative differences are consistent with the corresponding improvements observed in FID.

Considering the total operation count as diffusion steps times fixed-point iterations per step, even where the total is comparable, implicit solvers concentrate computation into fixed-point iterations that are cheaper on analog hardware, yielding a net energy advantage. Fig. 4C shows FID versus projected energy for different implicit solver configurations on the butterfly dataset compared to conventional explicit Euler sampling on digital hardware, each for 8, 16, 32, 64, 128 diffusion steps respectively. A detailed examination of these projections is deferred to the Discussion.

Figure S2 shows that implicit samplers approach convergence with as little as two fixed-point iterations; more iterations can improve the results especially for small numbers of diffusion steps but show diminishing improvement for 8 and more steps. In particular for eight or

more diffusion steps implicit sampling with two iterations yields similar or better results than a higher order Heun sampler, which is a commonly used best practice (*51*) and uses the same number of function evaluations. Additional qualitative results are presented in fig. S3, S4, S5.

Finally, Fig. 4D and fig. S6 qualitatively show that the implicit paradigm can be beneficial even for highly optimized state-of-the art models. We use FLUX.1 (*54*), a 13 billion parameter text-to-image diffusion model. As a rectified flow model optimized for few-step explicit Euler sampling, the model produces high quality results already at 5 explicit steps. To explore a possible advantage of implicit methods we thus reduce the number of steps to 2. The standard implicit scheme does not converge at such low step counts, as per previous discussion. However, Implicit-CN achieves robust fixed-point convergence: with 2 diffusion steps and 4 fixed-point iterations, it significantly improves sample quality, producing sharper fine-grained details. These initial findings underline the generality of the approach, extending beyond controlled experiments to industry-grade diffusion models.

Discussion

We introduce ADMs and experimentally demonstrate fully analog diffusion steps on an analog optical computer. ADMs exploit a correspondence between the equilibration dynamics of analog feedback systems and the fixed-point structure of implicit diffusion inference. This correspondence avoids the training burden typically associated with fixed-point models such as deep equilibrium models, while harnessing the energy efficiency of analog fixed-point dynamics. Efficiency is gained without redesigning the model and without forfeiting the pretrained weights, training pipelines, and continually improving algorithms of the mainstream ecosystem. We combine prototype hardware experiments with DT simulations to test algorithmic scalability beyond the present hardware scale. The projected end-to-end energy gain of ADMs on analog hardware, relative to explicit sampling on digital accelerators, depends on four factors: the per-operation hardware efficiency, the reduction in diffusion steps, the number of

fixed-point iterations per step, and, when latent-space scaling is used, the cost of a final digital decoder. Analog optical hardware is estimated to reach femtojoule-scale energies per operation (*12, 25–27*); here we use a conservative 100× advantage relative to current digital accelerators. Implicit sampling reduces the number of diffusion steps by up to 16× compared with explicit sampling, while each implicit step requires multiple fixed-point iterations, yielding a net 1-2× reduction in total network evaluations. For generation within the hardware's native resolution, where no digital decoder is required, the full 100× per-operation advantage should be preserved. For higher-resolution generation through standard latent-space scaling, a final digital decoding step is added; this decoder overhead reduces the projected end-to-end gain to 26× in our representative Smithsonian Butterflies setting (Fig. 4D). A detailed breakdown is provided in supplementary materials section B.

Realizing these projected energy gains for industrial and scientific applications requires analog hardware that matches the scale and precision requirements of modern generative models. Advances in low-bit and quantized inference on digital hardware (*55*) are expected to carry over to analog implementations, further increasing their efficiency. Scaling is particularly critical for high-dimensional image generation tasks, which demand millions or billions of weights. Significant progress has recently been demonstrated in this direction (*11, 14*), but planar optical approaches (*11, 14, 17, 25*) remain limited by chip reticle size, as routing and computation share the same plane. In contrast, the AOC hardware we used for the ADM implementation leverages the third dimension to separate computation from data routing, enabling improved scalability. Each SLM pixel can serve as an individual weight modulator, with a footprint that is several orders of magnitude smaller than in planar alternatives. Off-the-shelf SLMs support up to four million μm-scale pixels per panel, enabling million-weight modules that can be replicated toward the 100-million-weight regime. Emerging metasurface modulators, which promise 9.5 million pixels with microsecond-scale refresh rates (*53*), could extend this range further.

Diffractive systems achieve millions of parameters and have enabled impressive demonstra-

tions for tasks with fixed or infrequently updated weights (*10,56*). Workloads that require reprogrammable weights, including support for different models, pretrained weights, and an evolving algorithmic ecosystem call for a complementary form of generality. Vector-matrix architectures provide this generality by implementing matrix multiplication directly as a computational primitive, with weights that can be electronically updated and reused across models without refabrication. This capability requires a modulator for each weight and three-dimensional optical fan-in and fan-out for all-to-all connectivity, so reprogrammable systems, ours included, currently operate at smaller scale and in larger volumes (*19,26,27,57*). Our 2,304-weight system is representative of the currently demonstrated scale of this all-to-all connected reprogrammable regime. The central scaling challenge for this class of systems is therefore not simply how many elements can be accommodated, but whether the optical stack can be compacted while preserving the generality that motivates the architecture. Meeting it will require full integration within centimeter-scale volumes.

The modular design in Fig. 1A targets 100 million weights by distributing optical matrix-vector multiplications over replicated modules (see supplementary materials section B for scaling, latency and energy estimates). At this scale, the system supports below 58×58 RGB images directly in pixel space without requiring a digital decoder. Higher resolutions can be achieved through standard latent scaling, where all iterative denoising remains analog and only a final digital decoding step is added, as demonstrated in our Smithsonian Butterflies setting (256×256 RGB). The task-agnostic DT calibration procedure should scale without methodological changes (see supplementary materials Section A.3); the use of incoherent illumination and detection, which are inherently insensitive to phase drift and path-length variations (see supplementary fig. S15 for representative stability measurements), supports stability and reproducibility at scale. Measured hardware noise and DT simulations under corresponding noise conditions further show the noise-suppressing properties of fixed-point search and diffusion denoising (fig. S13). Finally, realizing full hardware efficiency requires analog memory such as a

sample-and-hold circuit to avoid digital conversions between diffusion steps.

Overall, the results presented here point toward a new trajectory for GenAI. As inference energy becomes a growing constraint on the deployment of AI at scale, we anticipate that aligning algorithmic structure with hardware physics, rather than adapting algorithms to hardware, may offer a general route to translating the per-operation efficiency of analog computing into system-level gains without sacrificing ecosystem compatibility.

References and Notes

1. J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, S. Ganguli, *International conference on machine learning* pp. 2256–2265 (2015).

2. J. Ho, A. Jain, P. Abbeel, *Advances in neural information processing systems* **33**, 6840 (2020).

3. J. Baldridge, *et al.*, *arXiv preprint arXiv:2408.07009* (2024).

4. A. Polyak, *et al.*, *arXiv preprint arXiv:2410.13720* (2024).

5. G. DeepMind, Veo 3, https://deepmind.google/models/veo/ (2024).

6. A. Blattmann, *et al.*, *arXiv preprint arXiv:2311.15127* (2023).

7. S. Lewis, *et al.*, *Science* (2025).

8. C. Zeni, *et al.*, *Nature* **639**, 624 (2025).

9. A. Ivanov, N. Dryden, T. Ben-Nun, S. Li, T. Hoefler, Data movement is all you need: A case study on optimizing transformers (2021).

10. Y. Chen, *et al.*, *Science* **390**, 1259 (2025).

11. S. R. Ahmed, *et al.*, *Nature* **640**, 368 (2025).

12. Y. Chen, *et al.*, *Nature* **623**, 48 (2023).

13. B. Dong, *et al.*, *Nature* **632**, 55 (2024).

14. S. Hua, *et al.*, *Nature* **640**, 361 (2025).

15. N. Leroux, *et al.*, *Nature Computational Science* pp. 1–12 (2025).

16. Y. Zhang, *et al.*, *Nature Photonics* pp. 1–7 (2025).

17. Y. Shen, *et al.*, *Nature photonics* **11**, 441 (2017).

18. I. Oguz, *et al.*, *Advances in Neural Information Processing Systems* **37**, 59150 (2024).

19. S. Chen, Y. Li, Y. Wang, H. Chen, A. Ozcan, *Nature* **644**, 903 (2025).

20. Y. Cheng, *et al.*, *Nature Communications* **16**, 5022 (2025).

21. J. Yang, *et al.*, *arXiv preprint arXiv:2404.05648* (2024).

22. B. J. Shastri, *et al.*, *Nature Photonics* **15**, 102 (2021).

23. J. Feldmann, N. Youngblood, C. D. Wright, H. Bhaskaran, W. H. Pernice, *Nature* **569**, 208 (2019).

24. G. Wetzstein, *et al.*, *Nature* **588**, 39 (2020).

25. R. Hamerly, L. Bernstein, A. Sludds, M. Soljačić, D. Englund, *Physical Review X* **9**, 021032 (2019).

26. T. Wang, *et al.*, *Nature Communications* **13**, 123 (2022).

27. K. P. Kalinin, *et al.*, *Nature* pp. 1–8 (2025).

28. S. Hooker, *Communications of the ACM* **64**, 58 (2021).

29. N. Al-Kayed, *et al.*, *Nature* **648**, 576 (2025).

30. T. Inagaki, *et al.*, *Science* **354**, 603 (2016).

31. T. Inagaki, *et al.*, *Nature Photonics* **10**, 415 (2016).

32. A. B. Klein, Z. Zhu, D. Saiham, G. Li, S. S. Pang, *Optics Letters* **49**, 194 (2024).

33. T. Honjo, *et al.*, *Science advances* **7**, eabh0952 (2021).

34. F. Böhm, G. Verschaffelt, G. Van der Sande, *Nature communications* **10**, 3538 (2019).

35. B. Wu, *et al.*, *Nature communications* **16**, 4296 (2025).

36. S. Bai, J. Z. Kolter, V. Koltun, *Advances in neural information processing systems* **32** (2019).

37. Z. Geng, A. Pokle, J. Z. Kolter, *Advances in Neural Information Processing Systems* **36**, 41914 (2023).

38. A. Pokle, Z. Geng, J. Z. Kolter, *Advances in Neural Information Processing Systems* **35**, 37975 (2022).

39. X. Bai, L. Melas-Kyriazi, Fixed point diffusion models (2024).

40. A. Momeni, *et al.*, *Nature* **645**, 53 (2025).

41. L. G. Wright, *et al.*, *Nature* **601**, 549 (2022).

42. M. J. Rasch, *et al.*, *Nature communications* **14**, 5282 (2023).

43. R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer pp. 10684–10695 (2022).

44. O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation (2015).

45. W. Peebles, S. Xie, Scalable diffusion models with transformers (2023).

46. Y. Song, *et al.*, *arXiv preprint arXiv:2011.13456* (2020).

47. Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, M. Le, *arXiv preprint arXiv:2210.02747* (2022).

48. X. Liu, C. Gong, Q. Liu, *arXiv preprint arXiv:2209.03003* (2022).

49. M. S. Albergo, E. Vanden-Eijnden, *arXiv preprint arXiv:2209.15571* (2022).

50. J. Song, C. Meng, S. Ermon, *arXiv preprint arXiv:2010.02502* (2020).

51. T. Karras, M. Aittala, T. Aila, S. Laine, *Advances in neural information processing systems* **35**, 26565 (2022).

52. A. Jolicoeur-Martineau, K. Li, R. Piché-Taillefer, T. Kachman, I. Mitliagkas, *arXiv preprint arXiv:2105.14080* (2021).

53. M. Shebanow, H. Finkelstein, P. Bowen, Revolutionizing the ai data center: Delivering 4,000 peta operations per second at 1% of the power using optical systolic arrays, *White paper*, Neurophos Inc. (2025). Accessed: 2026-01-28.

54. B. F. Labs, *et al.*, *arXiv preprint arXiv:2506.15742* (2025).

55. A. Gadhikar, R. Grazzi, J. Hensman, *et al.*, Optrot: Mitigating weight outliers via data-free rotations for post-training quantization (2024).

56. X. Lin, *et al.*, *Science* **361**, 1004 (2018).

57. Y. Liang, *et al.*, *Light: Science & Applications* **15**, 115 (2026).

58. X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks (2010).

59. I. O. Tolstikhin, *et al.*, *Advances in neural information processing systems* **34**, 24261 (2021).

Acknowledgments

We acknowledge colleagues at Microsoft Research, Cambridge, UK for fruitful discussions.

Author contributions

H.K., F.F., and J.C. contributed equally to this work. J.C. and F.P. are corresponding authors. J.G., F.P. and H.B. conceived and managed the project. H.K. and F.F. developed the algorithmic framework for the 2D experiments. H.K. and J.C. performed the digital-twin simulations for the 2D experiments. H.K. derived the ADM algorithm and theoretical results. H.K. and J.C. performed the large-scale simulations. F.F. performed the hardware simulations for the latent experiments. J.C., J.H.C., D.J.K., D.C., G.B. and F.P. conceived, designed and implemented the optical and analog electronic hardware. J.C. designed and performed the hardware experiments, analysed the resulting data and implemented the digital twin with support from J.H.C., D.J.K., D.C., G.B. and F.P. D.J.K., J.H.C., F.P., K.P.K. and D.C. developed the scaling roadmap. J.C., D.J.K. and J.H.C. developed the control software. B.R., K.P.K., F.P., C.G., P.J., B.C., M.H., S.R., V.R., J.K. and H.B. provided technical input to the simulations and experiments. H.K., J.C., F.P., F.F., J.H.C. and K.P.K wrote the paper with input from all authors.

Supplementary Materials

Materials and methods

A Analog hardware platform

A.1 Hardware implementation

Experiments were conducted on an analog optical computing platform comprising 2,304 programmable weights implemented through 3D optical matrix multipliers integrated with channel-wise analog electronic nonlinearities in a recursive architecture (fig. S1). A photograph of the analog optical hardware, used to generate all the experimental datasets is shown in fig. S1D, with key components highlighted in fig. S1B. A schematic diagram of the optical subsystem is provided in fig. S1C. The system comprises a microLED source array, two SLMs, two PD arrays, and an imaging system.

The platform operates through a sequence of fixed-point iterations, each comprising three steps: nonlinear activation, matrix multiplication, and summation with fixed bias, time-embedding, and conditioning voltages. The same weights are reused at every iteration, following an in-memory computing style. The system runs until the signals stabilize, reaching a fixed point for each respective diffusion step. Finally, the voltages are recorded using a 48-port digitizer (Spectrum M2p.5913). Each fixed-point iteration implements Equation 5. The term $\sigma(z_k)$ is the channel-wise tanh nonlinearity, realized in analog electronics by bipolar differential pairs. The product $W\sigma(z_k)$ is computed optically: $\sigma(z_k)$ drives the microLED array (encoding as light intensities), W is encoded on the SLM pixels, and the column-wise photodiode summation yields the matrix-vector product as voltages. The conditioning terms $b + te + pad(x)$ are injected as additive voltages, which is a constant vector per diffusion step. During each fixed-point iteration, the system state vector z_k switches back and forth between optical and analog electronic

forms.

MicroLED source array z_k transitions from analog electronic voltages to light intensities via a custom-fabricated microLED source array consisting of 48 independently addressable channels, each comprising four green emitters arranged in a 4 × 48 layout. The emitters are structurally identical to display-grade microLEDs and are operated in parallel within each channel to distribute optical power. The 4×48 layout is chosen to increase per-channel optical power, with four emitters fanning into each row of 48 weight elements. Images of the array under an inspection microscope and projected onto a camera are shown in fig. S1B and fig. S9C; a camera image of the LED spreader is shown in fig. S9D.

Optical matrix multiplier Light emitted by the sources (peak wavelength 503 nm at 20 mA) is collected by an objective lens and imaged onto the SLMs through a combined 4F system formed by the objective and lens groups. The weight matrix W is encoded onto reflective SLM pixels; light from each microLED fans out across a row of the SLM for element-wise multiplication, and the resulting signals are summed column-wise by a one of the 1 × 48 PD array as electrical voltages. Weights are trained as continuous float32 values with no quantization applied. Prior to hardware inference, a post-training quantization pipeline normalizes and relaxes the weights before encoding them as asymmetric 9-bit integer matrices for the hardware. The SLM plane has a pixel pitch of 8.5 μm and a fill factor of 92%; a camera image of the SLM plane is shown in fig. S9E. The microLED pitch at the SLM is 193 μm. The normalized response of one SLM as a function of gray value is reported on the top of fig. S9A; both SLMs exhibit comparable performance.

The present implementation introduces three key modifications relative to our prior platform (*27*). First, the spreading lens is replaced with a Thorlabs LJ1567L1-A to accommodate the approximately tenfold increasee in the number of weights. Second, the optical paths corresponding to positive and negative weights are spatially separated rather than interleaved, to

reduce optical crosstalk between the two paths. Third, a cylindrical merging lens (Thorlabs LJ1638L1-A) is placed in front of each PD array. This lens compresses the vertical extent of the SLM image on the detector while preserving its horizontal magnification. The image of the reflective SLM is then out of focus in the vertical direction, but this is not a problem as we only require summation of these signals.

Analog electronics The nonlinear activation is implemented using bipolar differential pairs. The custom electronic control circuitry, implemented using off-the-shelf surface-mount components on printed circuit boards, is described in detail in Supplementary Material B.2 (*27*). In the current prototype, the inter-step conditioning operation (a per-channel affine transform) is performed digitally between diffusion steps and applied as voltages injected at the subsequent diffusion step (corresponding to the additive conditioning term b+te+pad(x) in the ADM equation; see fig. 1B for a schematic). All iterative denoising updates within each diffusion step are executed in the analog domain. Fully analog state propagation across diffusion steps would require analog memory together with programmable gain and offset. Such analog memory can be implemented using a sample-and-hold circuit.

Bandwidth The end-to-end bandwidth of the optical subsystem, measured with the microLEDs driven at 10 mA, is reported in fig. S1B, using a Keysight Technologies E5061B network analyser. The tested signal was injected at the LED driver board, propagated through the full optical subsystem, and detected at the PD end. The PD output was routed to a breakout board for the measurement. The -3 dB cutoff point occurs at 5 MHz, which reflects the combined response of the microLEDs, PDs, and associated electronic circuitry, and sets the upper bound for analogue iteration speed in the current hardware configuration.

A.2 Sampling and experiment workflow

Waveforms were measured with a sampling rate of 4.6875 MHz. Each experiment captured 90 samples, corresponding to a total acquisition window of 19.2 µs. The fixed-point search remained entirely in the analog domain during this interval. Samples were collected between 68% and 93% of the solve time, which translates to a time window from 13.1 µs to 19.7 µs. This range ensures that equilibrium dynamics are captured after initial transients but before full convergence. Once sampled, readout voltages were linearly converted to solution variables. Voltages were averaged over 100 experiment for figures in the main text; averaging over 5 experiments shows comparative generation quality (see fig. S7 for 2D distributions and fig. S8CD for latent tasks).

A.3 DT calibration procedure and accuracy

The DT is calibrated in two stages: an open-loop calibration that identifies component-level hardware parameters, and a closed-loop calibration that refines system-level dynamics. In the open-loop stage, we measure the behavior of each physical block—electrical offsets and noise characteristics, the analog nonlinearity, microLED input efficiency, SLM response, photodiode gain and crosstalk, and polarization-beam-splitter leakage—using dedicated experiments designed to isolate each contribution. Together, these component-level effects comprise the principal hardware nonidealities discussed in the ADM methodology; characterizing them enables defining the initial parametric form of the DT. To capture remaining linear distortions, we perform an open-loop OVMM sweep over 100 random weight matrices, 6 input vectors, and 7 values of the gradient gain β (which scales the optical signal applied to the $\mathrm{W}\sigma(z)$ term in Equation 5), and fit output-correction vectors. Across 4200 open-loop experiments, the mean NRMSE was 0.015 (fig. S10A). This modular calibration workflow is general in structure and is intended to be adaptable, in principle, to other analog computing platforms comprising linear-nonlinear building blocks. Each channel is characterized through automated experiments that

sweep input and output channels independently, with the weight matrix partially or fully activated to isolate individual channel responses. Since the sweep drives input and output channels independently, the characterization procedure generalizes to larger systems without methodological changes. Per-channel DT accuracy is a property of the hardware platform rather than the model or workload. The incoherent LED sources preclude inter-channel optical interference, making it relatively stable and reproducible across operating conditions (see fig. S15 for representative stability measurements). This suggests that per-channel DT accuracy is not expected to degrade as the hardware scales up.

In the closed-loop stage, we validate and adjust the DT under recurrent operation. Using 1000 experiments that span random weight matrices, channel-wise nonlinear coefficients, and realistic β configurations, we refine a small set of global scaling parameters: the gains on the gradient, input, and the α skip connection term described in Equation 10, along with residual offsets. This yields a digital model that accurately reproduces the hardware's equilibrium trajectories across configurations used in the diffusion experiments. Mean NRMSE across 1000 closed-loop experiments was 0.009 (fig. S10B).

To further quantify accuracy during diffusion inference, we measure the per-step NRMSE for each diffusion step by comparing the hardware output of a single denoising update against its corresponding DT evaluation (fig. S10C for 2D tasks, and fig. S10E for latent-space tasks). Aggregating results across multiple 2D diffusion tasks yields a mean per-step NRMSE of 0.013, while across multiple latent-space experiments achieve a mean NRMSE of 0.01.

Because the diffusion model is trained under the assumption of ideal correspondence between analog and digital execution small per-step discrepancies accumulate along the sampling trajectory. We therefore also report the accumulated NRMSE relative to the DT as a function of diffusion step index fig. S10D and fig. S10F. Although the accumulated error increases with depth, it remains within a range compatible with successful generation across all 2D and latent benchmarks.

Finally, we observe systematic differences between the error distributions in 2D and latent experiments. This likely reflects the general calibration strategy of the DT, whose parameters are fitted using random weight matrices rather than optimized for task-specific operating regimes, promoting generalization across workloads.

A.4 Sensitivity to DT-hardware mismatch

To assess how much DT–hardware mismatch can be tolerated before inference quality degrades, we perform a perturbation study on the digital twin. We apply multiplicative perturbations of magnitude ε in [0, 20%] independently to four error modes: the skip connection gain α, the gradient gain β, the inter-step conditioning vector $b+te+pad(x)$, and a per-iteration output gain. In each case the perturbation is applied uniformly across all 48 channels as a multiplicative factor $(1 \pm \varepsilon)$. Figure S12 summarizes the results. The system is most sensitive to iteration gain and conditioning perturbations, while α and β perturbations produce comparatively modest degradation. An asymmetry is observed between positive and negative perturbations: amplification $(+\varepsilon)$ erodes the contraction mapping property and can cause divergence, whereas attenuation $(-\varepsilon)$ preserves contractivity and degrades output quality more gradually.

A.5 Noise

Noise in the hardware arises from LED relative-intensity noise, photodetector shot and thermal noise, electronic dark noise, and temporal fluctuations in the optical weights. Among these, SLM phase flicker dominates, causing time-varying perturbations to the effective weight matrix. fig. S9A shows the measured output-voltage distributions across grey levels in an open loop experiment. Since this measurement is taken at the end of the optical subsystem, primarily driven by the SLM flicker, and represents the effective noise injected into the computation. We further measure the hardware noise across gain settings, with each noise distribution characterized from 1,120,000 samples (fig. S13A). The observed range defines the per-iteration noise

levels used in the DT simulations.

We then evaluate inference under corresponding Gaussian perturbations. Using an exemplar 2D distribution, the DT simulations provide intuition for how noise affects implicit diffusion inference. The comparison between diffusion step 0 and the final diffusion step suggests that the diffusion process itself provides some tolerance to per-iteration noise. The same simulations also illustrate a compute–noise trade-off: increasing post-convergence fixed-point averaging reduces NRMSE, allowing higher noise levels to be tolerated at the cost of additional analog iterations (fig. S13B,C).

We next evaluate the implications of noise for image generation and energy cost using Smithsonian Butterflies, Oxford Flowers-102, and AFHQ on DT. For each dataset, architecture, and noise level, we first sweep the diffusion-step count until 128 steps for explicit sampling and use the best FID obtained as the digital baseline. We then sweep both the number of implicit diffusion steps and the number of fixed-point iterations per step. Increasing the number of fixed-point iterations generally improves FID, consistent with iterative refinement toward the fixed point, but this improvement comes at an energy cost. We therefore identify operating regions that maintain competitive FID while preserving projected analog-hardware energy-efficiency gain, with the energy model described in supplementary materials section B. Across datasets and architectures, the preferred operating point depends on both the noise level and the decoder/backbone cost: some settings retain substantial energy gain with little degradation in FID, whereas others require more fixed-point iterations and therefore reduce the end-to-end advantage (fig. S13D,E).

Consistent with these simulations, experimental trial-wise averaging increases generation quality. Fig. S7A shows that averaging over five repeats improves the per-step NRMSE from 4.1×10^{-2} to 1.9×10^{-2} for 2D experiments, and negaligible improvment beyond 5 averages for all datasets (fig. S7A and fig. S8CD). Because the dominant noise source in this case is SLM flicker, the averaging cost is not fundamental. If measurements are synchronized to the SLM

flicker, its temporal waveform could be incorporated into the DT, potentially reducing the need for experimental averaging.

B Scaling, latency and energy estimations

For a module realizing weight count N, because the accumulate operations are performed passively in the optical domain, while the main power-consuming optoelectronic components are the $\sqrt{N}$ source drivers and $\sqrt{N}$ detector channels, the dominant energy per operation scales as $N^{-1/2}$.

The prototype used here implements N=2,304 programmable weights, each weight encoded on multiple SLM pixels. The architecture scales along two axes. First, on hardware, the per-module weight count N is bounded by the number of addressable pixels of the SLM, so a single module can grow toward 4M weights using off-the-shelf modulator arrays. A modular design reaches larger models by partitioning the vector-matrix multiplication into blocks, assigning each block to one of M identical modules and combining the partial sums in analog electronics, so that $N_{\mathrm{tot}} = MN/2$, where the factor of two accounts for encoding positive and negative weights separately. (*27*) proposes a N_{tot}=100M system comprising N=4M weights per module and M=50 modules. The 2000 input variables in such settings are targeted at 2cm die size and 2GHz speed, corresponding to tens of nanoseconds timescale per fixed-point search. Total power estimating accountering for optics, electronics, analog-to-digital conversion and digital-to-analog conversion, as well as memory read/write operations is estimated to be in the order of 806.5W, hence 495 TOPS/W. For comparison, an NVIDIA A100 provides 2.08 TOPS/W for dense INT8 tensor operations and 4.16 TOPS/W with structured sparsity, using the 300 W power envelope. The GPU simulations in this work use FP16, for which the corresponding peak efficiency is 1.04 TFLOPS/W. We therefore use a conservative normalized efficiency ratio of $\epsilon_{hw}/\epsilon_{gpu}$=100, where ϵ_{hw} and ϵ_{gpu} are the corresponding energy efficiencies of the analog hardware and GPU.

To estimate the supported image resolution, we consider a neural-network implementation in which each layer is distributed across multiple hardware modules or mapped within one SLM displaying a tiled block-matrix structure (Fig. 1A). We denote the ratio of activated weights by η. For a K-layer network with input/output dimension d_z and hidden width d_i, the number of active weights can be approximated as $\eta N_{tot} = 2d_z d_i + (K-2)d_i^2$, hence $d_z = (\eta N_{tot} - (K-2)d_i^2)/2d_i$. Let R be the side length of the generated image in pixels, C_{in} the number of generated channels, and c the output-space expansion factor. The supported resolution satisfies $R^2 C_{in} \leq c d_z$, and therefore $R^2 \leq c(\eta N_{tot} - (K-2)d_i^2)/2d_i C_{in}$. For example, the no-compression case is c=1, and with η=1, C_{in}=3, N_{tot}=100M, the maximum possible latent dimension is d_z=10,000, giving $R <$58.

Standard latent-space operation provides a second route for higher image resolutions. A diffusion model acting in the latent space of a VAE with per-axis spatial downsampling factor f and C_z latent channels processes vectors whose dimension is reduced by $c = f^2 C_{\mathrm{in}}/C_z$ relative to pixel space, with $C_{\mathrm{in}} = 3$ for RGB images (*43*). The value of c depends on the autoencoder. The frozen Stable Diffusion VAE (f=8, C_z=4) gives c=48 (*6*), the FLUX VAE ($f = 8$, $C_z = 16$) gives $c = 12$ (*54*). In our Smithsonian Butterflies generation setting, c=48 and N_{tot}=58M. The Mixer-L/4 operates on a d_z=4,096-dimensional latent ($C_z = 4$, spatial 32×32), which is decoded into 256×256 RGB images. The fixed-point iterations still run on analog hardware, and realizing this latent leverage adds a single digital decoding step at the end.

When c=1 and no final decoding layer is used, the energy efficiency of the analog hardware is fully preserved. If a final digital decoding layer is used to improve generation capability, the end-to-end energy consumption is $E_{hw,e2e} = F_{denoise}/\epsilon_{hw} + F_{decode}/\epsilon_{gpu}$, where $F_{denoise}$ and F_{decode} are operation counts required for denoising and decoding, respectively. We define $\rho = F_{decode}/F_{denoise}$, where $F_{denoise} = T n_{fp} n_{tok} \eta N_{tot}$, T is the number of diffusion steps, n_{fp} is the number of fixed-point iterations per diffusion step, and n_{tok} is the architecture multiplier, defined as the number of multiply-accumulate operations per forward pass normalized by N_{tot}.

The effective energy efficiency that considers this final digital decoder is $(1+\rho)/(1/\epsilon_{hw}+\rho/\epsilon_{gpu})$. Thus, ρ quantifies the penalty from the digital decoding stage: when ρ approaches 0, the end-to-end efficiency gain approaches raw analog hardware efficiency gain, whereas when ρ is large, the digital decoder dominates. The decoding workload is $F_{decode} = \gamma R^2$, where γ is the number of decoder multiply-accumulate operations per output pixel. Thus, the realized end-to-end gain is architecture dependent. The decoder factor γ is set by the decoding architecture: for example, the Stable Diffusion VAE requires approximately 9.46M FLOPs per output pixel, whereas TAESD requires approximately 0.539M FLOPs per output pixel. The factor n_{tok} scales with the sequence length of the denoising model; for example, n_{tok}=256 for DiT-S/2, 64 for Mixer-L/4, and 1 for a plain multilayer perceptron. Across various settings, ρ can range from 0.01 to order unity, so the digital decoder can be either a small correction or the dominant bottleneck. In our Smithsonian Butterflies generation setting, using a TAESD decoder and Mixer-L/4 denoiser with N_{tot}=58M, n_{tok}=64, we obtain $\rho \approx$0.037 for the explicit baseline at diffusion step T_b=128, which gives FID=28.0. The implicit setting reaches comparable quality, FID=27.8, with 16 diffusion steps and two iterations per step, yielding an effective gain of $(T_b + T_b\rho)/(Tn_{fp}\epsilon_{gpu}/\epsilon_{hw} + T_b\rho) \approx$26.

C Flow matching and definitions

Sampling from a flow-matching based diffusion model corresponds to integrating the ODE

$$\frac{d}{dt}x(t) = f_\theta(x(t), t), \qquad x(0) = x_0, \tag{1}$$

from $t = 0$ to $t = 1$, where $x_0 \sim N(0, I)$ is a sample from a multi-variate Gaussian and in our case $f_\theta : \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}^d$ denotes a time-conditional, K-layer feedforward neural network of

the form

$$f_\theta(x,t) = \big(L_K \circ \cdots \circ L_1\big)(x,t), \qquad L_i(x,t) = W_i\sigma(x) + b_i + te_i. \tag{2}$$

Each layer is a map $L_i : \mathbb{R}^{d_{i-1}} \times \mathbb{R}_+ \to \mathbb{R}^d$, with weight $W_i \in \mathbb{R}^{d_i \times d_{i-1}}$, bias $b_i \in \mathbb{R}^{d_i}$, and time embedding vector $e_i \in \mathbb{R}^{d_i}$, $i = 1, \ldots, K$, and $d_0 = d_K = d$. The trainable model parameters are $\theta := \{W_1, b_1, e_1, \ldots, W_K, b_K, e_K\}$. A common ODE solver is the explicit Euler discretization, which for a fixed number of steps $T = \frac{1}{\Delta t}$ iterates the update

$$x_{t+\Delta t} = x_t + \Delta t\, f_\theta(x_t, t). \tag{3}$$

In contrast the implicit Euler scheme uses the implicit update rule

$$x_{t+\Delta t} = x_t + \Delta t\, f_\theta(x_{t+\Delta t}, t + \Delta t), \tag{4}$$

which, as $x_{t+\Delta t}$ appears on both sides, requires solving a system of non-linear equations at each step. Importantly, both updates use the same operator $\mathrm{Euler}_{x_t,t,\Delta t,\theta}(z) := x_t + \Delta t\ f(z,t;\theta)$ merely evaluated at different $t \in [0,1]$ and $z \in \mathbb{R}^d$.

D ADMs: Methodology

An analog feedback system relaxes to a physical equilibrium, and an implicit diffusion step is likewise specified by a fixed-point condition. ADMs make this correspondence practical by embedding the network in Equation 2 into a higher-dimensional analog state space $\mathbb{R}^n$, where the computation is implemented by repeated application of a fixed update map. The map is evaluated explicitly during training (Equation 3) and implicitly by a fixed-point search during

inference (Equation 4) We denote by

$$\mathrm{ADM}_{x,t;\theta}(z) = W\sigma(z) + b + te + \mathrm{pad}(x) \tag{5}$$

the ADM operator, a re-parameterization of the update implemented by AOC, where $z \in \mathbb{R}^n$ is the internal AOC state, $b, e \in \mathbb{R}^n$ are concatenations of biases b_i and time embeddings e_i respectively, and $\mathrm{pad}(x) := (x, 0, \ldots, 0)^\top \in \mathbb{R}^n$ embeds the diffusion state $x \in \mathbb{R}^d$ into the analog state space $\mathbb{R}^n$. AOC iterates this update until convergence to a fixed-point $z^* \in \mathbb{R}^n$, with $\mathrm{ADM}_{x,t;\theta}(z^*) = z^*$.

The purpose of the digital twin is to incorporate the non-idealities expected at inference time during training. However, this makes it crucial that training and inference rely on the same update operator. The below construction of the ADM operator ensures this property.

Blockmatrix layout Multi-layer ADMs can be implemented either combining multiple AOC systems in a modular way (Fig. 1A), where all modules participate in a shared recurrent loop, or by using a single AOC system with the block structure shown in the bottom of Fig. 1A (implicit). Both implementations are mathematically equivalent and we will focus on the latter in the following.

The block-structured matrix W and bias b of ADMs are constructed such that successive applications of the ADM operator correspond to cycling the signal through the layers of the network. Specifically, with each application of the ADM operator the signal gets propagated down the state and the output of the last layer gets fed back to the top. For $\Delta t = \frac{1}{T}$ we define

$\theta(\Delta t) = \{W_{\Delta t}, b_{\Delta t}, e_{\Delta t}\}$ with

$$W_{\Delta t} = \begin{bmatrix} 0 & 0 & \cdots & 0 & \Delta t\, W_K \\ W_1 & 0 & \cdots & 0 & 0 \\ 0 & W_2 & \ddots & 0 & 0 \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & 0 & \cdots & W_{K-1} & 0 \end{bmatrix}, \quad b_{\Delta t} = \begin{bmatrix} \Delta t\, b_K \\ b_1 \\ b_2 \\ \vdots \\ b_{K-1} \end{bmatrix}, \quad e_{\Delta t} = \begin{bmatrix} \Delta t\, e_K \\ e_1 \\ e_2 \\ \vdots \\ e_{K-1} \end{bmatrix}, \tag{6}$$

as well as $\mathrm{pad}(x) = (x, 0, \ldots, 0)^\top \in \mathbb{R}^n$.

Training Setting $\Delta t = 1$ at training time, after K applications of the ADM operator, the first d components (defined as $[\cdot]_{:d}$) of the AOC state recover the network output. In contrast to evaluation in the hardware which always iterates the operator until convergence to a fixed-point, during training on GPU using the DT we can evaluate the system after a finite number of ADM iterations. This means at training time we can evaluate the model explicitly via

$$f_\theta(x, t) = \left[(\mathrm{ADM}_{x=0,t;\theta(1)})^K(x) \right]_{:d}, \tag{7}$$

where ADM^K denotes K applications of the ADM operator, and thus use standard (explicit) training procedures.

Using this formulation, model parameters θ are learned using standard conditional flow matching with optimal transport path (*47*). Given samples $x_0 \sim p_0$ (Gaussian base distribution) and $x_1 \sim p_1$ (data distribution), as well as timestep $t \sim \mathrm{Uniform}([0, 1])$ and the linear interpolation $x_t = tx_1 + (1 - t)x_0$, the training objective is

$$\mathcal{L}_{\mathrm{CFM}} = \mathbb{E}_{(x_0,x_1),t} \left\| f_\theta(x_t, t) - (x_1 - x_0) \right\|_2^2, \tag{8}$$

with f_θ as defined in Equation 7.

Inference To correctly account for the non-idealities of the hardware, at inference time we can only apply the same operator learned during training. By construction of the ADM operator, K applications of $\mathrm{ADM}_{x_t,t;\theta(\Delta t)}$ on x_t yield the explicit Euler update Equation 3, i.e.,

$$\left[\mathrm{ADM}^{K}_{x_t,t;\theta(\Delta t)}(x_t)\right]_{:d} = \mathrm{Euler}_{x_t,t,\Delta t;\theta}(x_t) = x_t + \Delta t\ f_\theta(x_t, t). \tag{9}$$

The underlying working principle of ADMs is that iterating this update rule (with $t \to t + \Delta t$) beyond K iterations is still meaningful as the fixed-point of the Euler operator provides the implicit Euler update Equation 4 and as the Euler operator itself can be represented by K ADM iterations, consequently the fixed-point of the ADM operator can be identified with the implicit Euler update. This is formalized by the following theorem.

Theorem 1 (ADM Inference)**.** *Let σ be $\mathrm{Lip}(\sigma)$-Lipschitz and let $\|\cdot\|_\infty$ denote the induced matrix norm. If $\Delta t < \frac{1}{\mathrm{Lip}(\sigma)^K \|W\|_\infty^K}$, then for any (x, t) the operator $\mathrm{ADM}_{x,t;\theta(\Delta t)}$ has a unique fixed point $z^* \in \mathbb{R}^n$ and, for any initialization z_0, we have $\lim_{k\to\infty} \mathrm{ADM}^k_{x,t;\theta(\Delta t)}(z_0) = z^*$. Moreover, the readout components $[z^*]_{:d}$ of the fixed-point solve the implicit Euler equation Equation 4.*

Details and proofs can be found in *Supplementary Section G*.

Finally, combining these aspects yields our training and inference process described in Algorithm 1.

The same duality between explicit and implicit Euler scheme can be directly applied to make the Crank-Nicolson solver, whose update rule is given by

$$x_{t+\Delta t} = x_t + \Delta t/2\,(f_\theta(x_t, t) + f_\theta(x_{t+\Delta t}, t + \Delta t))\,,$$

compatible with analog hardware. To this end, first evaluate $\tilde{x}_t = x_t + \Delta t/2\ f_\theta(x_t, t)$ either in digital or using our analog version of explicit diffusion sampling (see *Supplementary Section G.2*). Using this pre-computed $\tilde{x}_t$, the result of the Crank-Nicolson scheme is given by

the fixed-point of the operator $\mathrm{ADM}_{\tilde{x}_t,t+\Delta t;\theta(\Delta t/2)}$, which is immediately compatible with AOC. Note that convergence of the Crank-Nicolson solver is already guaranteed for the less restrictive condition $\Delta t < \frac{2}{\mathrm{Lip}(\sigma)^K \|W\|_\infty^K}$, which translates to a 2× smaller minimum number of steps and allows more robust few-step sampling as demonstrated by the results on AFHQ in Fig. 4.

E Relaxed fixed-point iterations to address hardware constraints

Optical vector–matrix multiplication (OVMM) requires fanning-out and fanning-in optical signals to and from all the modulators, i.e., the pixels of the SLM in AOC, representing the weights. As matrices become increasingly sparse, the total reflected or transmitted optical power decreases, making the system progressively more lossy. In the extreme case of an identity matrix, all optical power is attenuated except at a single modulator per row and column, rendering such operations impractical in the optical domain. To solve this general optical hardware limitation, we propose a novel methodology that is implemented in AOC as follows. We implement an analog skip connection that bypasses the OVMM for a fraction α of the signal, while the remaining $(1-\alpha)$ fraction is processed through the lossy optical path. During training, such a skip connection would break the duality between explicit and implicit Euler schemes that underlies ADMs. However, when applied as a post-processing step to a trained model, this mechanism corresponds to a relaxed fixed-point iteration and can be exploited without modifying the training procedure. Specifically, we define an α-relaxed fixed-point iteration as

$$\mathrm{ADM}^{(\alpha)}_{x,t;\theta}(z) := \alpha \odot z + (1-\alpha) \odot \mathrm{ADM}_{x,t;\theta}(z), \qquad 0 \le \alpha < 1, \tag{10}$$

which introduces an analog skip connection while preserving the structure of the ADM operator. Crucially, after convergence the relaxed operator provides the same result as the original ADM

operator, since the mappings $x \mapsto f(x)$ and $x \mapsto \alpha x + (1-\alpha)f(x)$ admit identical fixed-points. Under the same step-size conditions as in the non-relaxed case, the α-relaxed operator converges to a unique fixed point corresponding to the diffusion update (Theorem 6). The skip connection can therefore be introduced post-training through a rescaling of the learned model parameters and only increases the number of fixed-point iterations required for convergence.

By selecting α appropriately, either globally or in a channel-wise (per analog state variable) manner, we can trade off optical power loss against the number of fixed-point iterations. This enables the representation of large-weight matrices within the dynamic range of the hardware while maintaining stable convergence and an improved effective signal-to-noise ratio. Further details are provided in *Supplementary Section G.3*.

F Simulation details on the digital twin

F.1 Analog Diffusion models (ADMs) on two-dimensional (2D) tasks

For the generation of the 2D distributions reported in Fig. 2, we use a four-layer fully connected model with input and output dimension 2 and hidden dimensions $[15, 16, 13]$. All models are initialized using Xavier initialization (*58*), after which weights are rescaled by a factor of 71.19. This factor is derived as described in *Supplementary Section G.4*, where the scaling parameters $s_1 = 0.0028$, $s_2 = 4.993$, and $s_3 = 0.0$ are fitted to the AOC activation function.

Models are trained using AdamW with weight decay 10^{-4}, a learning rate of 5×10^{-3}, and a cosine learning-rate schedule with linear warm-up over the first 500 steps. Training is performed for 20,000 steps with a batch size of 256 on 50,000 synthetically generated data points. To construct pairs of data and noise samples, we use the exact optimal transport solver from the POT Python package. Timesteps are sampled uniformly from $[0, 1]$, and no loss reweighting is applied.

F.2 Latent ADMs

For the results shown in Fig. 3, we employ a latent diffusion approach (*43*) in which generative modeling is performed in the latent space of a pretrained variational autoencoder (VAE), while diffusion inference is carried out using an analog diffusion model. The VAE is trained digitally, and its parameters are frozen prior to diffusion training.

In the first stage, the VAE is trained to reconstruct images from a low-dimensional latent representation by minimizing

$$\mathcal{L}_{\mathrm{VAE}} = -\mathbb{E}_{z\sim q_\phi(z|x)}\big[\log p_\theta(x|z)\big] + \epsilon\,\mathrm{KL}(q_\phi(z|x)\,\|\,p(z))\,, \tag{11}$$

where $p(z) = \mathcal{N}(0, I)$ denotes a standard Gaussian prior, $q_\phi(z|x)$ is a Gaussian posterior with diagonal covariance parameterized by an encoder network g_ϕ, and $p_\theta(x|z)$ is a Gaussian likelihood parameterized by a decoder network f_θ. We use a small regularization weight $\epsilon = 10^{-6}$ and do not include perceptual or adversarial loss terms. The latent dimensionality is set to $\dim(z) = 10$, which provides reasonable reconstruction fidelity.

The encoder and decoder are convolutional ResNet architectures with three resolution levels (32, 64, and 128 channels), each consisting of two residual blocks. Each block comprises three 3×3 convolutional layers with batch normalization and SiLU activations. Downsampling and upsampling are implemented via average pooling and nearest-neighbor interpolation, respectively. The VAE is trained using Adam with a batch size of 128, a learning rate of 10^{-3}, and no weight decay. All latent diffusion experiments use identical training hyperparameters across datasets.

For the latent ADM, we use a four-layer model with input and output dimension 10 and hidden dimensions $[12, 13, 13]$. Initialization and training follow the same protocol as in the 2D experiments described in *Supplementary Section F.1*.

F.3 ADMs at Scale on the digital twin

All experiments use a latent approach based on the Stable Diffusion VAE (Hugging Face implementation with tag `stabilityai/sd-vae-ft-mse`) which is kept frozen and is not fine-tuned for the specific tasks. We train all models using the standard flow matching formulation with optimal transport path (*47*). No loss weighting or rectification is applied. All models are trained with AdamW for 300,000 steps with a batch size of 64, learning rate 10^{-4} and weight decay 10^{-2}. All implicit methods use plain fixed-point iterations to compute fixed-points. More advanced solvers, like Broyden's method can marginally improve the results for the smallest numbers of diffusion steps but are omitted here for simplicity. The fixed-point iteration is initialized at the explicit Euler update which is taken into account as 1 fixed-point iteration, in particular for the energy calculations.

Digital Twin (DT) & Butterflies The experiment uses the Smithsonian Butterfly dataset at a resolution of 256×256, with horizontal flips as data augmentation. The digital twin of our hardware is restricted to (i) feedforward (MLP) layers, (ii) $\tanh$ activations, and (iii) skip connections. To satisfy these constraints, we adopt the MLP-Mixer architecture (*59*), which, analogously to transformers, processes images via patchification followed by alternating MLPs over the patch and channel dimensions. To remain as faithful as possible to the hardware constraints, all normalization layers (e.g., LayerNorm) are removed. We employ sinusoidal time embeddings and learned positional encodings. The model consists of 24 blocks. Each block comprises a two-layer MLP with hidden dimension 768 and skip connections for both the patch- and channel-mixing stages respectively, resulting in a total of 58,086,208 parameters. Due to the small size of the dataset (1000 samples) we train on the whole dataset and evaluate FID and KID using the train set as common practice in generative modeling.

Diffusion Transformer & AFHQ We use the AFHQv2 dataset, comprising the categories cats, dogs, and wild animals, at resolution 256×256 with horizontal flips as data augmentation.

We employ a standard Diffusion Transformer (DiT) implementation, specifically DiT-S/2, consisting of 12 transformer blocks with hidden dimension 384, 6 attention heads, and a patch size of 2. The total number of model parameters is 32,572,816. Metrics are computed on a held-out test set comprising 20% of the total data.

U-Net & Oxford Flowers-102 We use the Oxford Flowers-102 dataset, square-cropped and resized to a resolution of 256×256. We adopt the standard U-Net architecture as implemented in the Hugging Face `UNet2DModel`, with 2 ResNet layers per block, 4 downsampling and upsampling stages, and output channel dimensions $[128, 256, 512, 512]$. Cross-attention is applied at the two lowest resolution levels. The total number of model parameters is 106,382,340. Metrics are computed on a held-out test set comprising 20% of the total data.

FLUX.1 We use the FLUX.1 model (Huggingface version with tag black-forest-labs/FLUX.1-schnell) and override its sampler to implement implicit solvers. For our qualitative comparison we generate images of 512×512 resolution using different prompts and evaluate the default sampler (explicit Euler) as well as the Crank-Nicolson solver (Implicit-CN) using a maximum of 4 fixed-point iterations. Different to all other experiments, which use simple fixed-point iterations, here we employ a Broyden solver to compute fixed-points as due to the extremely low number of diffusion steps, simple fixed-point iterations became unstable even for Implicit-CN.

G ADMs: Additional Simulation Details

This section provides technical results and implementation details that complement the main-text Methods. We refer to Methods for definitions and notation. Here we provide: (i) a contraction-based convergence guarantee and proof for the fixed-point inference of ADMs, (ii) a finite-iteration convergence statement for explicit diffusion models on AOC, (iii) details on α-relaxed fixed-point iterations and how they enforce optical gain constraints (including channel-

wise relaxation), and (iv) the procedure to compile standard neural network models to the AOC DT.

G.1 Convergence of implicit fixed-point inference

Below we will make use of the following simple result.

Lemma 2. *Let $f : \mathbb{R}^n \to \mathbb{R}^m$ denote a K-layer neural network defined via composition as*

$$f(x) = (L_K \circ \ldots \circ L_1)(x), \quad L_i(x) = W_i\sigma(x) + b_i,\ i = 1, \ldots, K. \tag{12}$$

Then the Lipschitz constant $\mathrm{Lip}(f)$ of f is given as

$$\mathrm{Lip}(f) = \mathrm{Lip}(\sigma)^K \prod_{i=1}^{K} \|W_i\|_\infty. \tag{13}$$

Proof. The Lipschitz constant of every layer L_i can be estimated via

$$\|L_i(x) - L_i(y)\|_\infty = \|W_i\sigma(x) - W_i\sigma(y)\|_\infty \leq \|W_i\|_\infty \|\sigma(x) - \sigma(y)\|_\infty \tag{14}$$

$$\leq \|W_i\|_\infty \,\mathrm{Lip}(\sigma)\|x - y\|_\infty, \tag{15}$$

to be $\mathrm{Lip}(f) = \mathrm{Lip}(\sigma)\|W_i\|_\infty$. With that the result follows recursively. □

Consider the ADM operator $\mathrm{ADM}_{x,t;\theta(\Delta t)}$ defined in Methods (Equation 5 and Equation 6). The following lemma provides a sufficient condition for existence and uniqueness of a fixed point.

Lemma 3 (Fixed-point existence and convergence)**.** *Let σ be $\mathrm{Lip}(\sigma)$-Lipschitz and let $\|\cdot\|_\infty$ denote the induced matrix norm. If $\Delta t < \frac{1}{\mathrm{Lip}(\sigma)^K \|W\|_\infty^K}$, then for any (x, t) the operator $\mathrm{ADM}_{x,t;\theta(\Delta t)}$ has a unique fixed point $z^* \in \mathbb{R}^n$ and, for any initialization z_0, we have $\lim_{k\to\infty} \mathrm{ADM}^k_{x,t;\theta(\Delta t)}(z_0) = z^*$.*

Proof. Consider the ADM operator Equation 6 with $L_i(z_i) = W_i\sigma(z_i)+b_i+te_i+[\mathrm{pad}(x)]_{d_i:d_{i+1}}$ and $L_K^{\Delta t}(z_K) = \Delta t L_K(z_K)$. Consider any $z \in \mathbb{R}^n$ with structure $z = (z_1, \ldots, z_K)^T$, then after K applications of the ADM operator we obtain

$$\left(\mathrm{ADM}_{x,t;\theta}\right)^K(z) = \begin{pmatrix} L_K^{\Delta t} \circ \ldots \circ L_1(z_1) \\ L_1 \circ L_K^{\Delta t} \circ \ldots \circ L_2(z_2) \\ L_2 \circ L_1 \circ L_K^{\Delta t} \circ \ldots \circ L_3(z_3) \\ \vdots \\ L_{K-1} \circ L_{K-2} \circ \ldots \circ L_1 \circ L_K^{\Delta t}(z_K) \end{pmatrix} =: \begin{pmatrix} F_1 \\ F_2 \\ F_3 \\ \vdots \\ F_K \end{pmatrix}(z), \tag{16}$$

where each entry applies a permutation of the layers $\{L_1, \ldots, L_K^{\Delta t}\}$ of the network. By Lemma 2 we thus have $\mathrm{Lip}(F_j) = \mathrm{Lip}(\sigma)^K \Delta t \prod_{i=1}^K \|W_i\|_\infty$ for $j = 1, \ldots, K$. Note that the presence of Δt is due to the scaling of the last layer in Equation 6. Consider any $u, v \in \mathbb{R}^n$,

$$\begin{aligned}
&\left\|\left(\mathrm{ADM}_{x,t;\theta_{\Delta t}}\right)^K(u) - \left(\mathrm{ADM}_{x,t;\theta_{\Delta t}}\right)^K(v)\right\|_\infty && (17)\\
\leq &\left\|\left(\mathrm{ADM}_{x,t;\theta_{\Delta t}}\right)^K\right\|_{\mathrm{op}} \|u - v\|_\infty && (18)\\
= &\left(\max_i \mathrm{Lip}(F_i)\right) \|u - v\|_\infty && (19)\\
\leq &\mathrm{Lip}(\sigma)^K \Delta t \left(\prod_{i=1}^K \|W_i\|_\infty\right) \|u - v\|_\infty && (20)\\
\leq &\underbrace{\mathrm{Lip}(\sigma)^K \Delta t \|W\|_\infty^K}_{=\mathrm{Lip}(\mathrm{ADM}_{x,t;\theta_{\Delta t}})} \|u - v\|_\infty && (21)
\end{aligned}$$

Now with a recursion argument we have $\mathrm{Lip}\left(\mathrm{ADM}_{x,t;\theta_{\Delta t}}\right) = \mathrm{Lip}\left(\left(\mathrm{ADM}_{x,t;\theta_{\Delta t}}\right)^K\right)^{1/K} = (\Delta t)^{1/K} \mathrm{Lip}(\sigma)\|W\|_\infty$. Thus for $\Delta t < \frac{1}{\mathrm{Lip}(\sigma)^K \|W\|_\infty^K}$, the ADM operator becomes a contraction, i.e., $\mathrm{Lip}\left(\mathrm{ADM}_{x,t,\theta_{\Delta t}}\right) < 1$, and the result follows by Banach's fixed-point theorem. □

Given that a unique fixed-point exists the following result shows that it can be identified with the implicit Euler update.

Theorem 4 (Implicit ADM)**.** *Under the conditions of Lemma 3, the readout components* $[z^*]_{:d}$ *of the unique fixed point* z^* *of the ADM operator solve the implicit Euler equation, i.e.,*

$$[z^*]_{:d} = x_t + \Delta t \, f_\theta([z^*]_{:d}, t + \Delta t). \tag{22}$$

Proof. As the conditions of Lemma 3 are fulfilled, there exists $z^* \in \mathbb{R}^n$ such that $\mathrm{ADM}_{x_t,t;\theta_{\Delta t}}(z^*) = z^*$ and consequently $\left(\mathrm{ADM}_{x_t,t;\theta_{\Delta t}}\right)^K (z^*) = z^*$. Applying the ADM operator K times we obtain

$$\left(\mathrm{ADM}_{x_t,t;\theta_{\Delta t}}\right)^K (z^*) = \begin{pmatrix} L_K^{\Delta t} \circ \ldots \circ L_1(z_1^*) \\ L_1 \circ L_K^{\Delta t} \circ \ldots \circ L_2(z_2^*) \\ L_2 \circ L_1 \circ L_K^{\Delta t} \circ \ldots \circ L_3(z_3^*) \\ \vdots \\ L_{K-1} \circ L_{K-2} \circ \ldots \circ L_1 \circ L_K^{\Delta t}(z_K^*) \end{pmatrix} = \begin{pmatrix} \mathrm{Euler}_{x_t,\Delta t}(z_1^*) \\ z_2^* \\ z_3^* \\ \vdots \\ z_K^* \end{pmatrix} = \begin{pmatrix} z_1^* \\ z_2^* \\ z_3^* \\ \vdots \\ z_K^* \end{pmatrix} \tag{23}$$

where we defined $\mathrm{Euler}_{x_t,\Delta t}(z) := x_t + f(z)\Delta t$. Now the first blockentry reads $\mathrm{Euler}_{x_t,\Delta t}(z_1^*) = z_1^*$, which identifies $z_1^* = [z^*]_{:d}$ with the implicit Euler update $x_{t+\Delta t}$ as defined in Equation 4. Now with z^* being a fixed-point of the ADM operator, we have $\left[\left(\mathrm{ADM}_{x_t,t;\theta_{\Delta t}}\right)^k (z^*)\right]_{:d} = [z^*]_{:d} = x_{t+\Delta t}$ for any $k \in \mathbb{N}_0$. □

Theorem 1 follows directly by combining Lemma 3 with Theorem 4.

G.2 Explicit Models on analog hardware

In addition to ADMs, which implement an implicit ODE solver at inference time, we provide a formulation of conventional explicit ODE solvers which is immediately compatible with AOC and fixed-point search. In the following we refer to these models as explicit models. Explicit models use an alternative block structure in which a single forward pass through all layers is completed after exactly $K + 1$ AOC iterations. In this case, the fixed-point coincides with

the explicit network evaluation, and the last d components of the AOC state yield the output. Explicit models are obtained by placing the matrices corresponding to the layers on the off-diagonal. Analogous to the implicit version we define the parameters of the explicit model as $\theta^e(\Delta t) = \{W^e_{\Delta t}, b^e_{\Delta t}, e^e_{\Delta t}\}$, with

$$W^e_{\Delta t} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0_{d\times d} \\ W_1 & 0 & 0 & \cdots & 0 \\ 0 & W_2 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & \Delta t\, W_K & 0 \end{bmatrix}, \qquad b^e_{\Delta t} = \begin{bmatrix} 0 \\ b_1 \\ b_2 \\ \vdots \\ \Delta t\, b_K \end{bmatrix}, \qquad e^e_{\Delta t} = \begin{bmatrix} 0 \\ e_1 \\ e_2 \\ \vdots \\ \Delta t\, e_K \end{bmatrix}. \tag{24}$$

Evaluation of the model at training time differs from ADMs only in the readout components,

$$f_\theta(x,t) = \Big[(\mathrm{ADM}_{x=0,t;\theta^e(\Delta t=1)})^K(x)\Big]_{-d:}, \tag{25}$$

where $[\cdot]_{-d:}$ selects the last d components. Using an input embedding that injects the current iterate into both the top and readout blocks $\mathrm{pad}^e(x) = (x, 0, \ldots, 0, x)^\top \in \mathbb{R}^n$, the explicit Euler update is recovered from the last d components of the fixed point of $\mathrm{ADM}_{x,t;\theta^e(\Delta t)}(z) = W^e_{\Delta t}\sigma(z) + b^e_{\Delta t} + te^e_{\Delta t} + \mathrm{pad}^e(x)$, i.e. $[z^*]_{-d:}$, which is found after exactly $K+1$ iterations, which is formalized next.

Proposition 5 (Explicit Models)**.** *Let $z^{(k+1)} = \mathrm{ADM}_{x_t,t;\theta^e(\Delta t)}(z^{(k)})$ denote the explicit ADM iteration. Then for any initialization $z^{(0)}$ and any $K' \geq K+1$ we have $z^{(K')} = z^{(K+1)}$, and the readout satisfies*

$$[z^{(K')}]_{-d:} = x_t + \Delta t\, f_\theta(x_t, t), \tag{26}$$

i.e. the explicit Euler update.

Proof. By construction, each application propagates the signal one layer forward without feedback to the top block. After $K+1$ iterations, all blocks depend only on x_t, and a further appli-

cation does not change the state. The last block equals $x_t + \Delta t f_\theta(x_t, t)$ due to the injected copy of x_t in the last d components of $\mathrm{pad}^e(x_t)$. □

While explicit models reach a stationary state similar to ADMs, they lack the attractor dynamics of the implicit formulation: after $K + 1$ iterations the result gets continuously recomputed with different realizations of the zero-mean analog noise, which at first glance allows for averaging as in the implicit case. However, as the analog noise undergoes non-linear transformations during the K+1 iterations needed to reach the readout components, the total corruption is not necessarily centered at zero anymore, which limits the usefulness of averaging. In contrast, implicit models are self-correcting as each individual application of the AOC operator can be interpreted as a gradient descent step towards the fixed-point, corrupted by zero-mean analog noise. This corresponds to stochastic gradient descent, which has well-explored convergence properties.

G.3 Relaxed fixed-point iterations

The optical vector–matrix multiplication (OVMM) unit in the AOC represents weights using SLMs, which natively encode values in a bounded range. Using paired modulators for positive and negative weights allows matrices with entries in $[-1, 1]$ to be represented. General weight matrices $W \in \mathbb{R}^{n\times n}$ are therefore implemented by normalizing to unit range and applying an optical amplification factor $\beta = \|W\|_\infty$.

In practice, additional signal loss arises from the optical fan-out of light across SLM pixels, particularly for sparse or highly unbalanced matrices such as the identity, where most pixels block light. These effects require further amplification to maintain sufficient signal strength. However, large amplification degrades the signal-to-noise ratio, imposing a practical upper bound $\beta \leq \beta_{\max}$. To allow representing arbitrary models, this motivates the relaxed fixed-point formulation introduced in Methods, which preserves the fixed-point of the model while reducing the required optical gain,

$$\begin{aligned}\mathrm{ADM}^{\alpha}_{x_t,t;\theta}(z) &:= \alpha \odot z + (1-\alpha) \odot \mathrm{ADM}_{x_t,t;\theta}(z) \\ &= \alpha \odot z + W_\alpha \sigma(z) + b_\alpha + te_\alpha + x_t^\alpha,\end{aligned} \tag{27}$$

where $\alpha \in [0,1)^n$ is fixed and $W_\alpha = (I - \mathrm{diag}(\alpha))W$, $b_\alpha = (1-\alpha) \odot b$, $e_\alpha = (1-\alpha) \odot e$, $x_t^\alpha = (1-\alpha) \odot x_t$. This relaxation reduces the required effective OVMM gain without modifying the solution of the fixed-point equation.

Theorem 6. *For any $\alpha \in \mathbb{R}^n$ with $\alpha_i \in [0,1)$, $i = 1, \ldots, n$, the α-relaxed ADM operator converges under the same step-size conditions as the standard ADM operator. Moreover, both operators have exactly the same fixed points.*

Proof. If $z^* = \mathrm{ADM}_{x_t,t;\theta}(z^*)$, then $\mathrm{ADM}^{\alpha}_{x_t,t;\theta}(z^*) = \alpha \odot z^* + (1-\alpha) \odot z^* = z^*$. Conversely, if $z^* = \mathrm{ADM}^{\alpha}_{x_t,t;\theta}(z^*)$, then $(1-\alpha) \odot z^* = (1-\alpha) \odot \mathrm{ADM}_{x_t,t;\theta}(z^*)$, and since $1-\alpha$ has strictly positive entries, this implies $z^* = \mathrm{ADM}_{x_t,t;\theta}(z^*)$. Contractivity follows as ADM^{α} is a convex combination of the identity and $\mathrm{ADM}_{x_t,t;\theta}$, and thus satisfies the same contraction condition as the unrelaxed operator. □

For a vector $z \in \mathbb{R}^n$ let $\mathrm{diag}(z) \in \mathbb{R}^{n \times n}$ denote the square matrix with z on the diagonal. From $W_\alpha = (I - \mathrm{diag}(\alpha))W$, it follows that $\|W_\alpha\|_\infty \leq (1 - \alpha_{\min})\|W\|_\infty$. Hence, for a given $\beta_{\max}$, any matrix W with $\beta = \|W\|_\infty$ can be implemented by choosing $1 - \beta_{\max}/\beta \leq \alpha_i < 1$. Using a uniform relaxation $\alpha_i(\beta) = 1 - \beta_{\max}/\beta$ yields $W_\alpha = \beta_{\max} W_{\mathrm{norm}}$ and the relaxed operator

$$\mathrm{ADM}^{\alpha(\beta)}_{x_t,t;\theta}(z) = \alpha(\beta) \odot z + \beta_{\max} W_{\mathrm{norm}} \sigma(z) + b_{\alpha(\beta)} + te_{\alpha(\beta)} + x_t^{\alpha(\beta)}.$$

To further reduce optical power loss, we normalize W row-wise, yielding $W = \beta\ \mathrm{diag}(a)\, W_{\mathrm{norm,cw}}$. The channel-wise amplification factors a_i can be absorbed into a channel-wise relaxation by

choosing $1 - \alpha_i^{\mathrm{cw}} = 1/a_i$. Imposing the gain constraint $\beta_{\max}$ gives $\alpha_i^{\mathrm{cw}}(\beta) = 1 - \beta_{\max}/(a_i\beta)$ and the operator becomes

$$\mathrm{ADM}_{x_t,t;\theta}^{\alpha^{\mathrm{cw}}(\beta)}(z) = \alpha^{\mathrm{cw}}(\beta) \odot z + \beta_{\max} W_{\mathrm{norm,cw}} \sigma(z) + b_{\alpha^{\mathrm{cw}}(\beta)} + te_{\alpha^{\mathrm{cw}}(\beta)} + x_t^{\alpha^{\mathrm{cw}}(\beta)}.$$

This channel-wise relaxation substantially reduces OVMM power loss and improves the effective signal-to-noise ratio while preserving the ADM fixed point at the expense of slower fixed-point convergence.

G.4 Compiling neural networks to the AOC DT

The DT compiles neural network models expressed in a conventional sequential form into an AOC-compatible block-matrix representation (see Methods). While the DT can, in principle, represent the same computations as its digital counterpart, the activation function implemented in hardware differs from the standard hyperbolic tangent by input and output scalings and an additive offset. Since $\tanh$ is not scale invariant, these differences must be compensated to preserve the behavior of standard initializations.

The AOC implements the iterative update

$$\mathrm{ADM}_{x,t;\theta}(z) = W\,\sigma_{\mathrm{AOC}}(z) + b + te + x, \tag{28}$$

where $\sigma_{\mathrm{AOC}}(u) = s_1\sigma(s_2 u) + s_3$ is a scaled and shifted $\tanh$. A corresponding conventional digital neural network with $\tanh$ activations implements

$$f_{x,t;\theta}(z) = W\,\sigma(z) + b + te + x. \tag{29}$$

To leverage standard initialization and scaling heuristics developed for networks of the form Equation 29 (*58*), we introduce scaling factors s_1, s_2 and s_3 to match the fixed-points of AOC

and the conventional neural net.

Specifically, equivalence of the fixed points of Equation 28 and Equation 29 is ensured by the parameter rescaling

$$W_{\mathrm{AOC}} = \frac{1}{s_1 s_2} W, \qquad b_{\mathrm{AOC}} = \frac{1}{s_2} b - s_3, \qquad e_{\mathrm{AOC}} = \frac{1}{s_2} e, \qquad x_{\mathrm{AOC}} = \frac{1}{s_2} x, \tag{30}$$

with the corresponding output relation $z^* = s_2 z^*_{\mathrm{AOC}}$. Under this transformation, the DT fixed-point equation reduces exactly to the standard $\tanh$ fixed-point equation. As the output of each diffusion step serves as the input to the next, no intermediate rescaling is required between diffusion steps; scaling is applied only once, before the first and after the final diffusion step.

H Evaluation metrics

H.1 Normalized root mean squared error (NRMSE)

The NRMSE measures point-wise deviation between evaluated values z_k and reference values z_k^{ref} over N samples:

$$\mathrm{NRMSE} = \frac{\sqrt{\frac{1}{N}\sum_{k=1}^{N} \|z_k - z_k^{\mathrm{ref}}\|_2^2}}{\sqrt{\frac{1}{N}\sum_{k=1}^{N} \|z_k^{\mathrm{ref}}\|_2^2}}. \tag{31}$$

H.2 Maximum mean discrepancy (MMD)

MMD measures the distance between the distributions of evaluated values $\{z_k\}_{k=1}^N$ and reference values $\{z_l^{ref}\}_{l=1}^M$ over N evaluated values and M reference values by measuring the distance of their kernel mean embeddings in a reproducing kernel Hilbert space. We use a standard Gaussian kernel $\kappa(\mathbf{z}, \mathbf{z}')$.

$$\mathrm{MMD}^2 = \frac{1}{N(N-1)} \sum_{k \neq k'} \kappa(z_k, z_{k'}) + \frac{1}{M(M-1)} \sum_{\ell \neq \ell'} \kappa(z_\ell^{\mathrm{ref}}, z_{\ell'}^{\mathrm{ref}}) - \frac{2}{NM} \sum_{k=1}^{N} \sum_{\ell=1}^{M} \kappa(z_k, z_\ell^{\mathrm{ref}}). \tag{32}$$

We refer to MMD^2 as MMD in the text.

H.3 Wasserstein-2 (W2) distance

For probability distributions μ and ν on a metric space, the Wasserstein-2 distance $W_2(\mu, \nu)$ measures the minimal expected squared cost of transporting mass from μ to ν:

$$W_2^2(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \int \|x - y\|^2 \, \mathrm{d}\pi(x, y),$$

where π ranges over all joint distributions with marginals μ and ν. In practice, W_2^2 is computed by solving a discrete optimal transport linear program between empirical distributions with uniform sample weights and cost matrix $C_{ij} = \|x_i - y_j\|^2$, returning the minimum total transport cost. We use the POT Python package to compute W2.

H.4 Fréchet Inception Distance (FID)

Fréchet Inception Distance (FID) is a distributional similarity metric that quantifies the discrepancy between real and generated image distributions in the feature space of a pretrained Inception-v3 network. Specifically, deep features are extracted from an intermediate layer of Inception-v3 for both real and generated images, and each feature distribution is approximated by a multivariate Gaussian. The FID corresponds to the closed-form 2-Wasserstein (Fréchet) distance between these Gaussian distributions, thereby capturing differences in both the mean (image fidelity) and covariance (sample diversity) of the feature representations. Lower FID values indicate closer alignment between generated and real image distributions.

Algorithm 1 Training and inference of ADMs

Require: Network parameters $\theta_{\Delta t}$, step size Δt, total diffusion steps T
Require: (Digital twin of) ADM operator $\mathrm{ADM}_{x,t;\theta}$

Training (explicit evaluation using digital twin)
for each minibatch **do**
  Sample $x_0 \sim p_0$ (Gaussian), $x_1 \sim p_1$ (data)
  Sample $t \sim \mathcal{U}[0,1]$
  Interpolate $x_t \leftarrow t x_1 + (1-t)x_0$
  Evaluate network explicitly: ▷ K-layer model

$$f_\theta(x_t, t) \leftarrow \left[(\mathrm{ADM}_{x=0,t;\theta_{\Delta t=1}})^K (x_t) \right]_{:d}$$

▷ $[\cdot]_{-d:}$ for explicit
  Compute loss ▷ Standard CFM loss

$$\mathcal{L} \leftarrow \| f_\theta(x_t, t) - (x_1 - x_0) \|_2^2$$

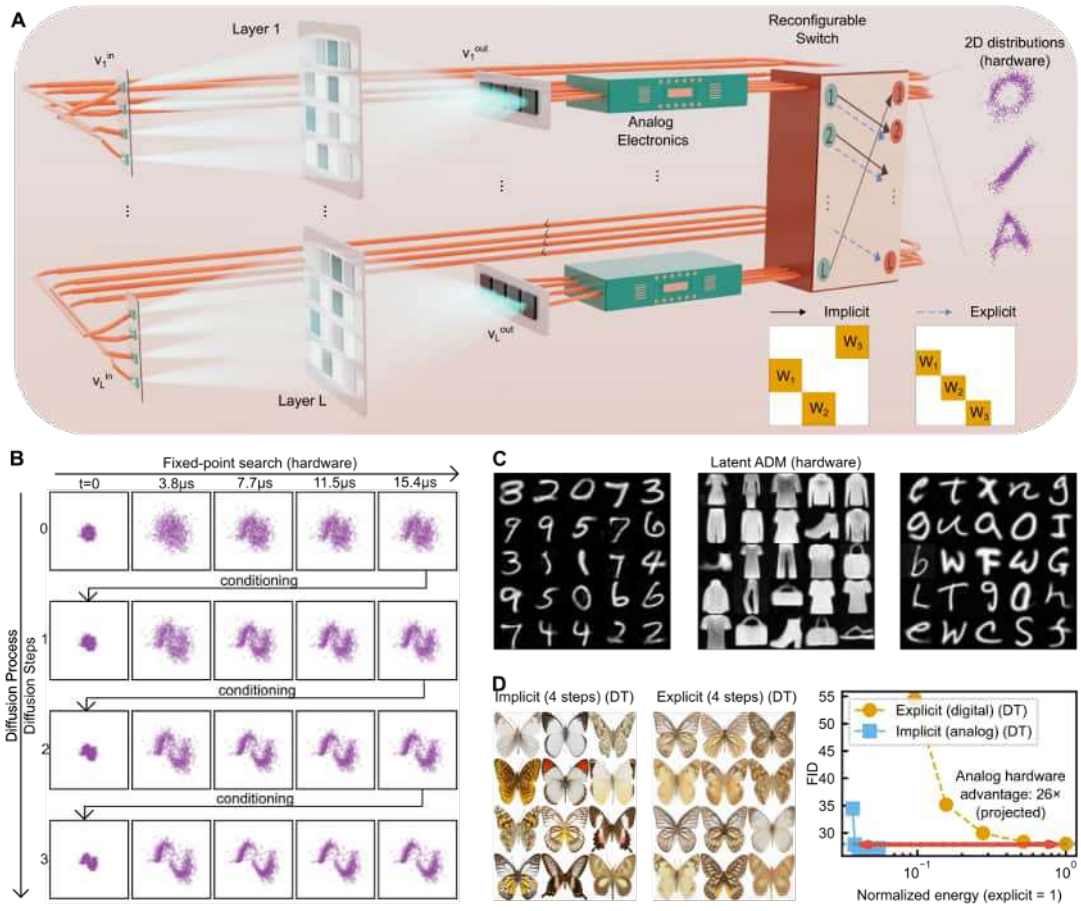
  Update parameters θ via backpropagation and gradient descent
end for

Inference (implicit evaluation using hardware):
Initialize $x_0 \sim p_0$
for $n = 1, \ldots, T$ **do**
  Set $t_n \leftarrow n/T$ ▷ $t_n \leftarrow (n-1)/T$ for explicit
  Initialize AOC state $z^{(0)}$ arbitrarily
  repeat
    $z^{(k+1)} \leftarrow \mathrm{ADM}_{x_n,t_n;\theta_{\Delta t}}(z^{(k)})$ ▷ Fixed-point search
  until convergence to fixed point z^* ▷ Theorem 1 guarantees convergence
  Extract denoised update: ▷ Take first d entries ($[\cdot]_{-d:}$ for explicit)

$$x_{n+1} \leftarrow [z^*]_{:d}$$

end for
return x_T

Figures



E

	Hardware today	DT at scale	Projected hardware
Task	2D distributions; MNIST, FMNIST, EMNIST	Smithsonian Butterflies; AFHQ; Oxford Flowers-102	Diffusion inference
Number of weights	2,304	32M-100M	100M
Number of color channels	1	3	3
Image resolution	2D distribution: 1 pixel latent experiments: 32 × 32	256 × 256	< 58 × 58 (no decoder); 256 × 256 (with decoder)
Latency for denoising	10-15 μs per diffusion step T=4 diffusion steps	Configurable number of iterations per step and number of diffusion steps	~10 ns per diffusion step; Configurable number of diffusion steps
Energy efficiency	<1GOPS/W	/	100× (no decoder); 26× (with decoder)

Fig. 1: Analog Diffusion Models (ADMs). **A,** ADMs implemented on fixed-point analog hardware. Multiple units are connected to implement the multi-layer structure of the diffusion model. A programmable switch selects between implicit and explicit evaluation modes. Mathematically, the multi-unit architecture is equivalent to a single system with a block-matrix layout, as illustrated in the inset, where each block encodes the weights of one layer and the block layout encodes the inter-layer connections. Each analog unit comprises optical vector–matrix

multipliers implemented using a three-dimensional optical system with spatial light modulators, with nonlinearities provided by analog electronics. Conditioning information is injected into the loop as fixed voltages, and the recursive system converges to a fixed point. Representative two-dimensional distributions generated by the hardware are shown on the right. **B,** ADM sampling on the AOC. Rows show diffusion steps, and columns show fixed-point search within each step, conditioned on the previous result. **C,** Selected latent ADM samples generated using the AOC for MNIST, FMNIST and EMNIST. **D,** Implicit (analog) versus explicit (digital) diffusion on Smithsonian Butterflies, simulated with the hardware digital twin (DT). Qualitative samples at 4 diffusion steps and FID versus projected energy for the two solvers. Each implicit step needs multiple fixed-point iterations, but fewer diffusion steps are required, so total operations are conserved. For reference, we assumed that analog hardware is 100× per-operation more efficient accounting for optics, electronics, ADC and DAC conversion, as well as memory read/write operations (*27*); the projected application-level gain is reduced by the energy cost of the digital final decoding step (see supplementary material section B). **E,** Scaling roadmap to larger systems. Left: the present analog hardware contains 2,304 programmable weights, at a scale comparable to current fully programmable, all-to-all connected free-space optical matrix engines (*19, 26, 27, 57*) and is used here to validate the operating principle. Middle: DT studies extend beyond the capacity of the present to test the algorithmic scalability. Right: a system-level hardware roadmap then estimates the image resolutions and effective end-to-end energy efficiencies achievable as the programmable optical architecture is scaled.

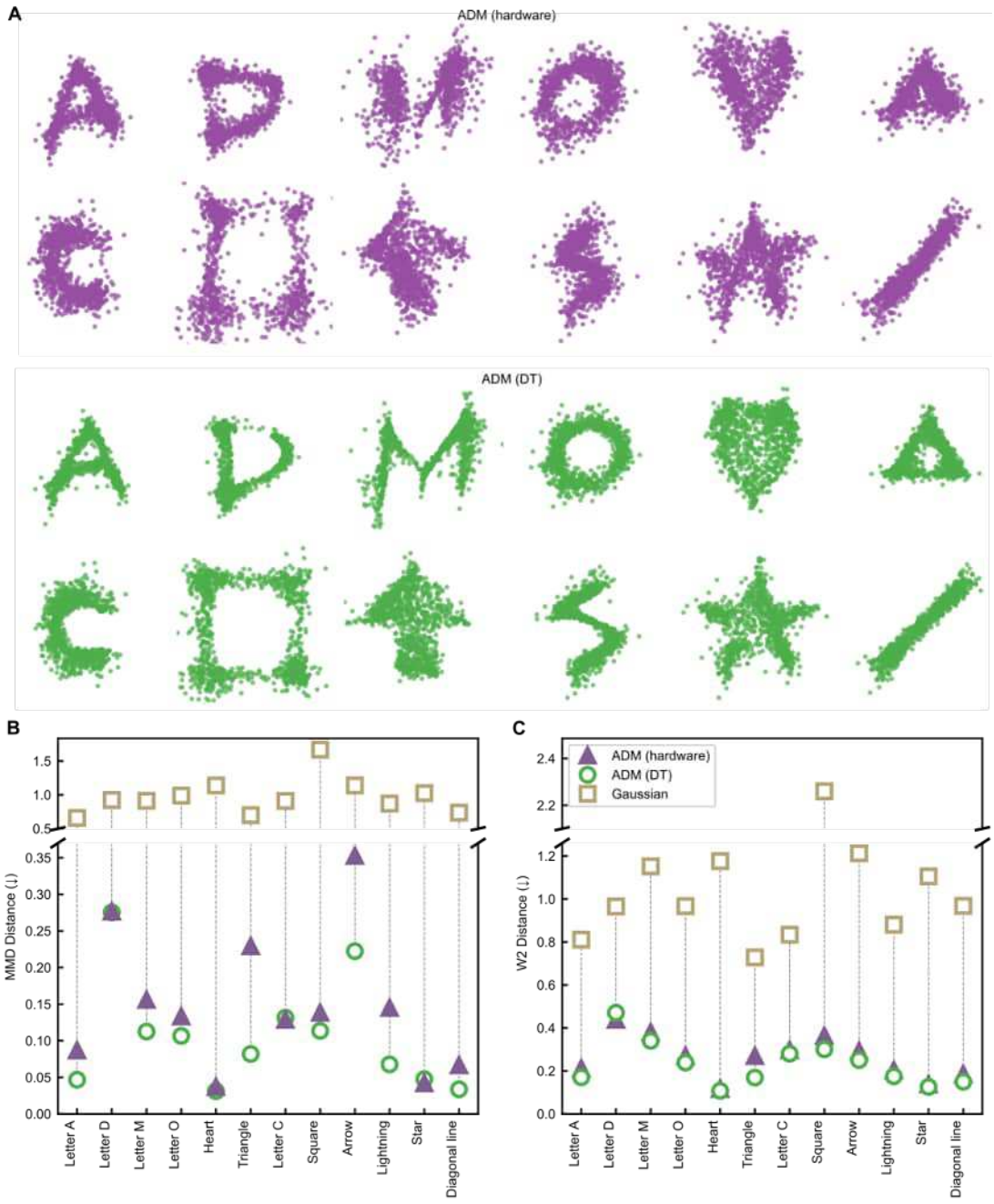


Fig. 2: ADM inference on hardware for 2D distributions. **A,** Examples of 2D distributions generated by ADMs on the hardware (top two rows) and its DT (bottom two rows). From left to right, we show: A, D, M, O, heart, triangle (top row), C, square, arrow, lightning, star and diagonal line (bottom row). **B,** Maximum Mean Discrepancy (MMD) between the groundtruth and distributions generated by the hardware, the DT and a Gaussian reference distribution. The reference distribution is obtained by fitting a 2D multivariate Gaussian to the dataset points. All 2D point clouds are normalized to the square $[-1, 1]^2$ using the dataset distribution for each shape as reference. **C,** Wasserstein-2 (W2) distance between the groundtruth and generated distributions for multiple 2D datasets.

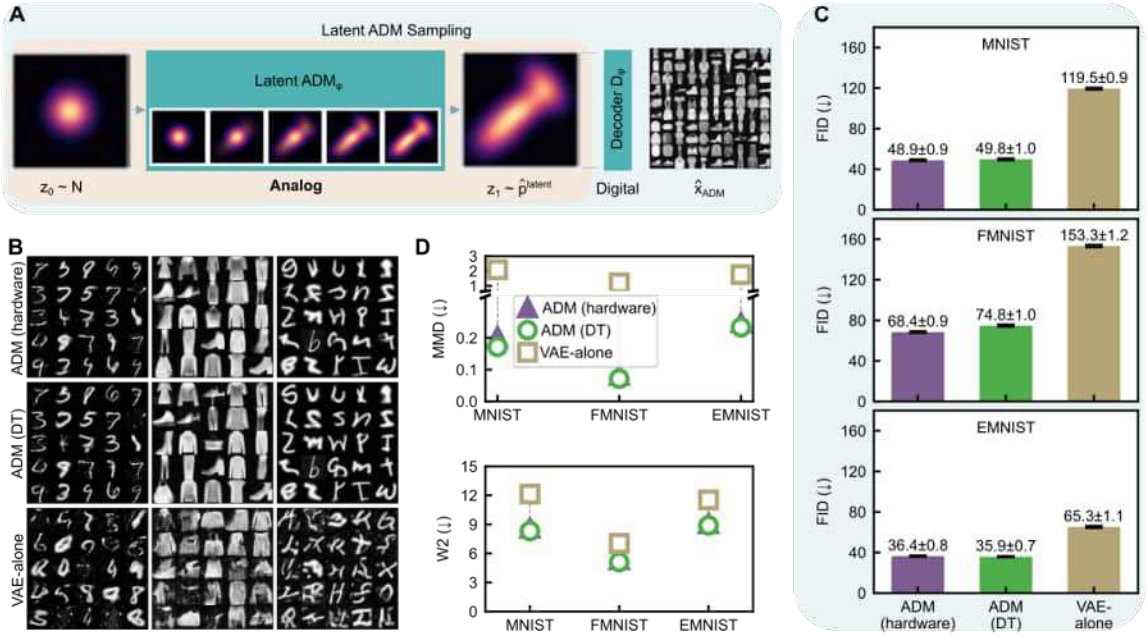


Fig. 3: Latent ADMs on hardware: operating principle, characterization and results. **A,** Schematic of the latent ADM framework. All diffusion steps are fully computed in analog hardware, and only the final decoding step is performed digitally. **B,** Samples generated by the latent ADM for MNIST (left), FMNIST (middle) and EMNIST (right), comparing ADMs on hardware (top), ADMs on the DT (middle) and the VAE-alone samples (bottom). **C,** Fréchet Inception Distance (FID) for MNIST (top), FMNIST (middle) and EMNIST (bottom), computed using 1,000 generated samples, comparing ADMs on hardware, ADMs on the DT, and the VAE-alone, highlighting the contribution of the ADM to the image generation. Error bars demote ±1 standard deviation over 100 evaluations with different reference subsets drawn from the training set. **D,** Latent-space distribution similarity between generated and target data (MMD and W2). Hardware results (triangles) closely match DT (circles), while VAE-alone (squares) serves as a baseline.

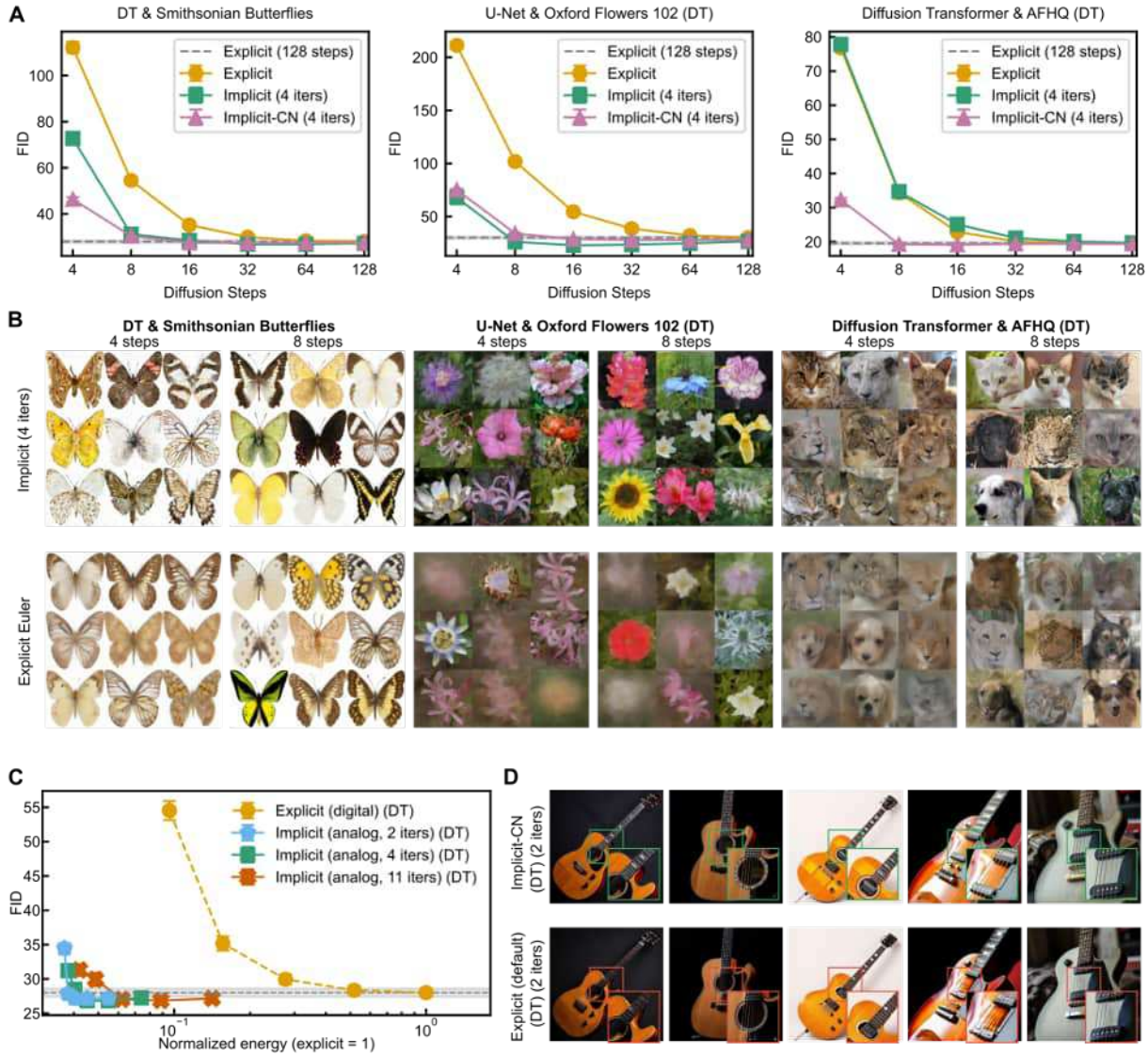
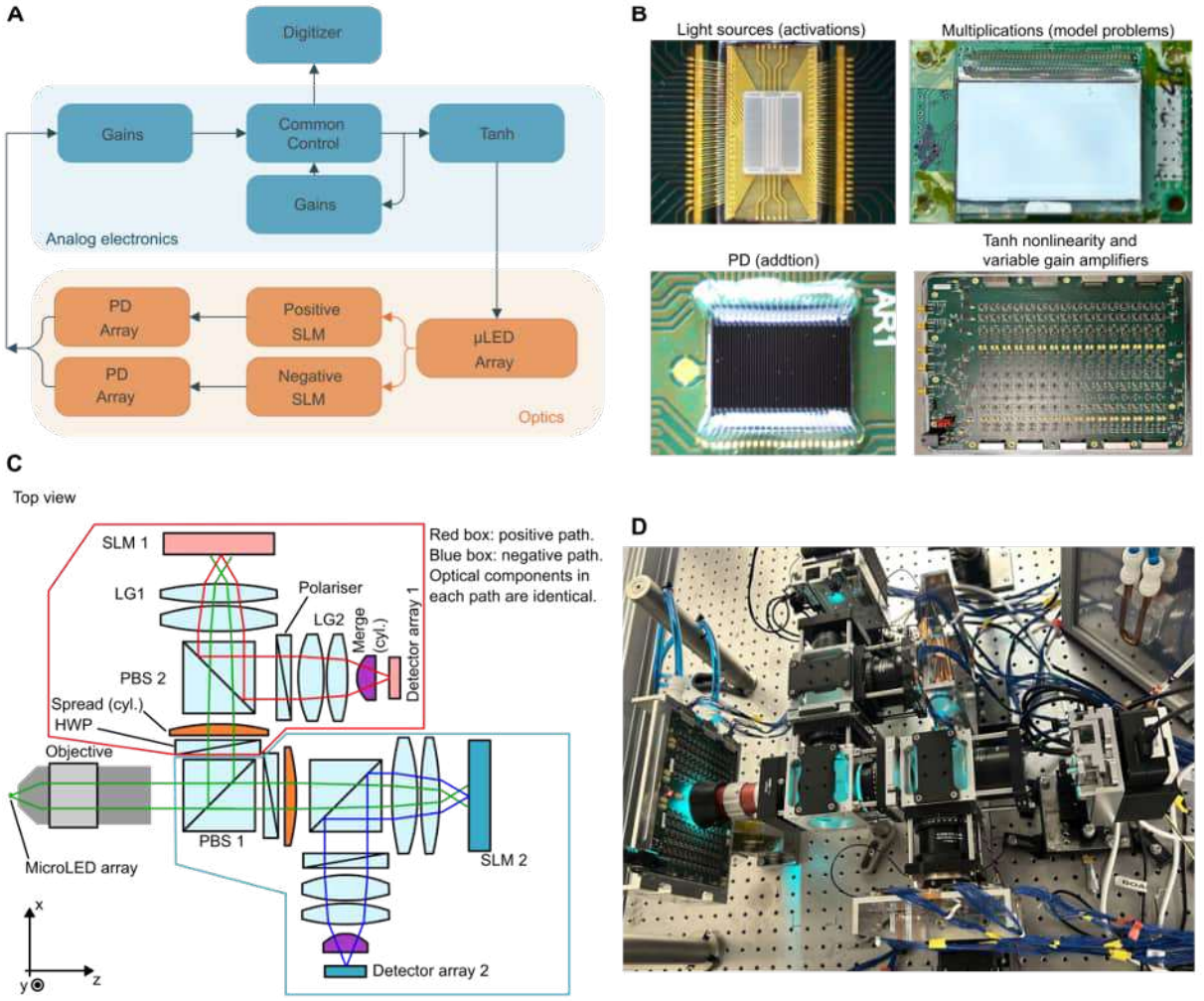


Fig. 4: Implicit sampling benchmarks across architectures and datasets (DT simulation). FID values are shown as mean ± standard deviation over 20 generations. **A,** FID versus number of diffusion steps for variants of implicit (this work) and explicit (standard) sampling from the same diffusion models for multiple datasets and model architectures. Implicit models use 4 fixed-point iterations per diffusion step. **B,** Qualitative comparison of sampling schemes at low step numbers. Butterflies (DT) and Flowers (U-Net) are generated with the standard Implicit method and AFHQ (DiT) shows the Implicit-CN variant. **C,** Trade-off study between sample quality and energy consumption: each curve represents the FID at 8, 16, 32, 64 and 128 diffusion steps respectively, versus the normalized energy for different numbers of fixed-point iterations per diffusion step. For reference, we assumed that analog hardware is 100× per-operation more efficient (*27*). **D,** Qualitative demonstration of Implicit-CN sampling with the state-of-the-art text-to-image diffusion model FLUX.1 (*54*) using the prompt "a guitar".

Supplementary Figures



E

	Nonlinearity	Vector-matrix multiplication	Gains and skip connection	Conditioning injection	VAE decoding (for latent experiments in Fig. 3 only)
Operation	$\sigma(z)$ in Equation 5	$W\sigma(z)$ in Equation 5	Skip-connection gain α in Equation 10 and gradient gain β described in fig. S2A	b + te + pad(x) in Equation 5	$p_\theta(x,z)$ in Equation 11
Domain	Analog electronic	Optical	Analog electronic	Analog electronic	Digital
Implementation	Bipolar differential pair	Imaging system (fig.S1C)	Variable gain amplifier	DAC-injected voltages	Pre-trained VAE on GPU

Fig. S1: Hardware architecture of the analog computing platform and corresponding key components. **A,** Schematic diagram of the analog electronic (in blue) and the optical (in orange) sub-systems, implementing the fixed-point update rule (Eq.1). Arrows are shown to indicate the signals' flow. **B,** Key hardware components photos. The 4x48 microLED array (top left) is the light source and represents neural network activations. The spatial light modulator (SLM) (top right) stores neural network weights, and multiplies them with the incoming light. The 48 photodetector array (bottom left) adds and transfers optical signals into the analog electronic domain. The nonlinearity, programmable variable gains, and other computations are applied in analog electronics. **C,** Schematic of the optical subsystem, illustrating dual optical paths for positive and negative weights. The sources, collected by the objectives, are imaged onto the

SLM using the combined 4F system of the objective and lens groups, respectively. **D,** Overhead view of the optical analog hardware. **E,** Summary of operations performed during inference. The VAE decoding column applies only to latent-space experiments (Fig. 3).

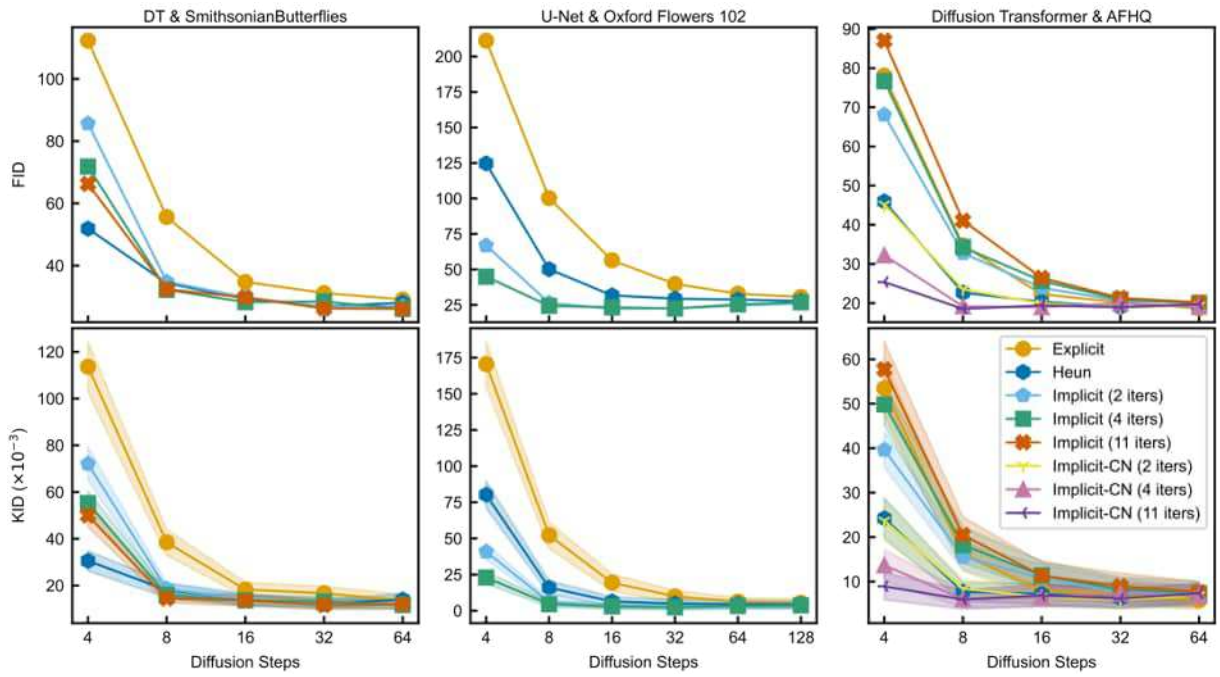


Fig. S2: FID and Kernel Inception Distance (KID) for additional sampler configurations. For 8 or more diffusion steps 2 fixed-point iterations can be sufficient to reach convergence of the implicit solvers. Implicit sampling can yield the same (on Butterflies with ≥ 8 steps) or better performance (on Flowers) with the same number of neural network evaluations compared to a higher order Heun solver, which is a common best practice (*51*). Implicit-CN with 2 fixed-point iterations is mathematically identical to the Heun scheme and additional iterations can provide further improvements as shown for AFHQ. In all experiments we use a warm start of the fixed-point search, by initializing at the explicit Euler result, which requires one network evaluation and hence is counted as one fixed-point iteration.

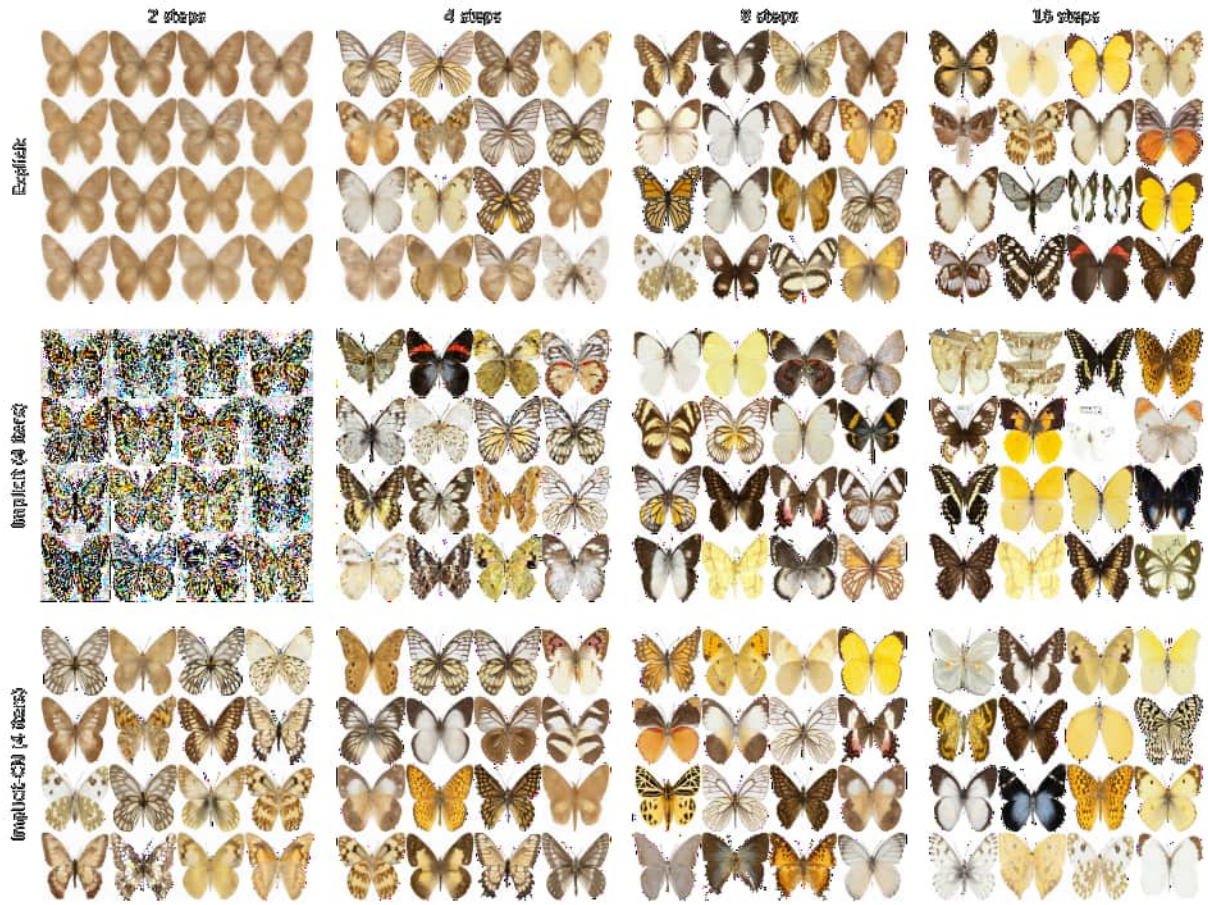


Fig. S3: Additional samples for the three schemes (Explicit, Implicit and Implicit-CN) from the Digital Twin trained on the Smithsonian Butterflies dataset.



Fig. S4: Additional samples for the three schemes (Explicit, Implicit and Implicit-CN) from the U-Net trained on Oxford-Flowers102.



Fig. S5: Additional samples for the three schemes (Explicit, Implicit and Implicit-CN) from the Diffusion Transformer (DiT) trained on AFHQ.

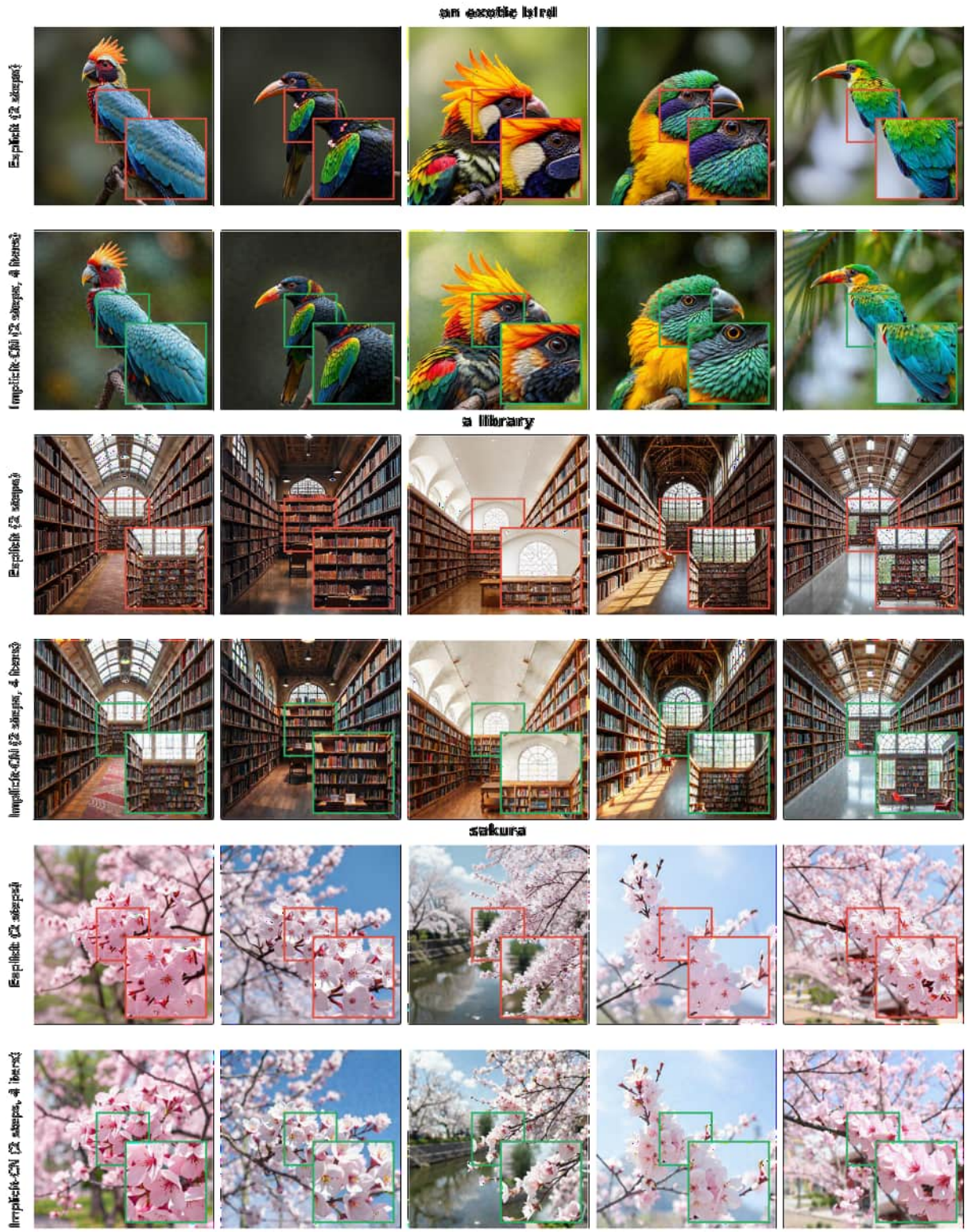


Fig. S6: Additional samples for different prompts produced with the state-of-the-art text-to-image diffusion model FLUX.1 (*54*) using explicit Euler (default) and Implicit-CN sampling with 2 diffusion steps and 4 fixed-point iterations. The respective prompts are contained in the titles.

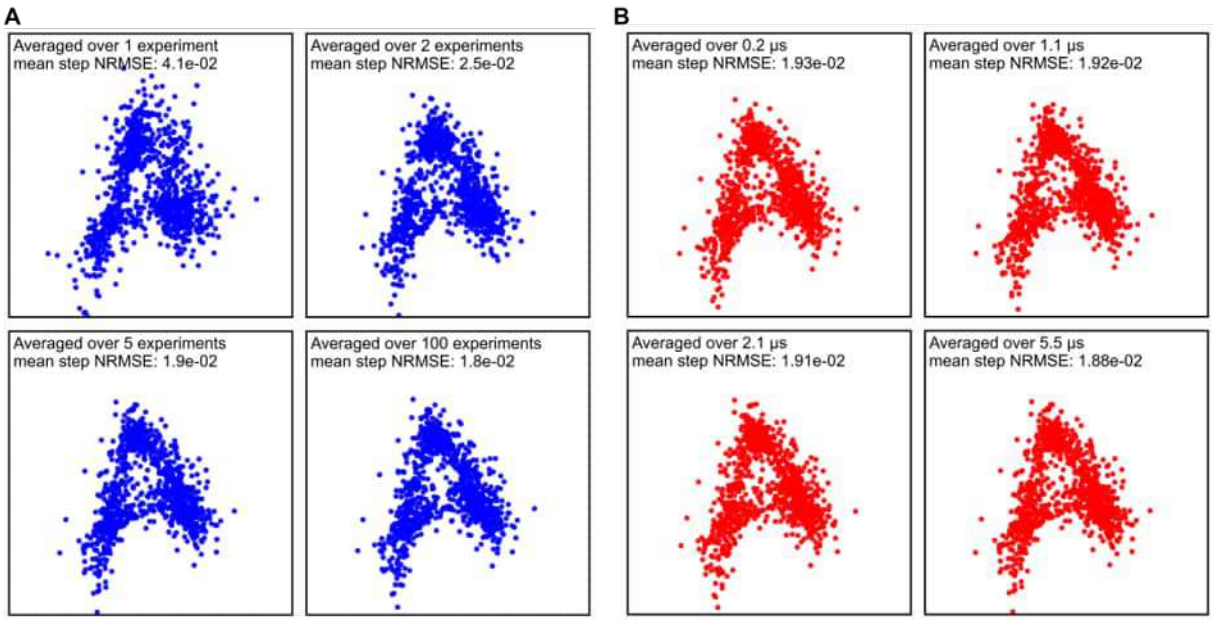


Fig. S7: Additional noise details on the 2D experiments. **A,** Effect of averaging across repeated 2D experiments for the letter A, chosen as a representative example, for ADMs generated on AOC. Increasing the averaging beyond 5 yields negiligible additional reductions in NRMSE. **B,** Effect of averaging within a fixed-point time window for the letter A for ADMs generated on AOC. Increasing the averaging window (up to 5.5 µs) yields small but consistent reductions in NRMSE, consistent with SLM-dominated noise.

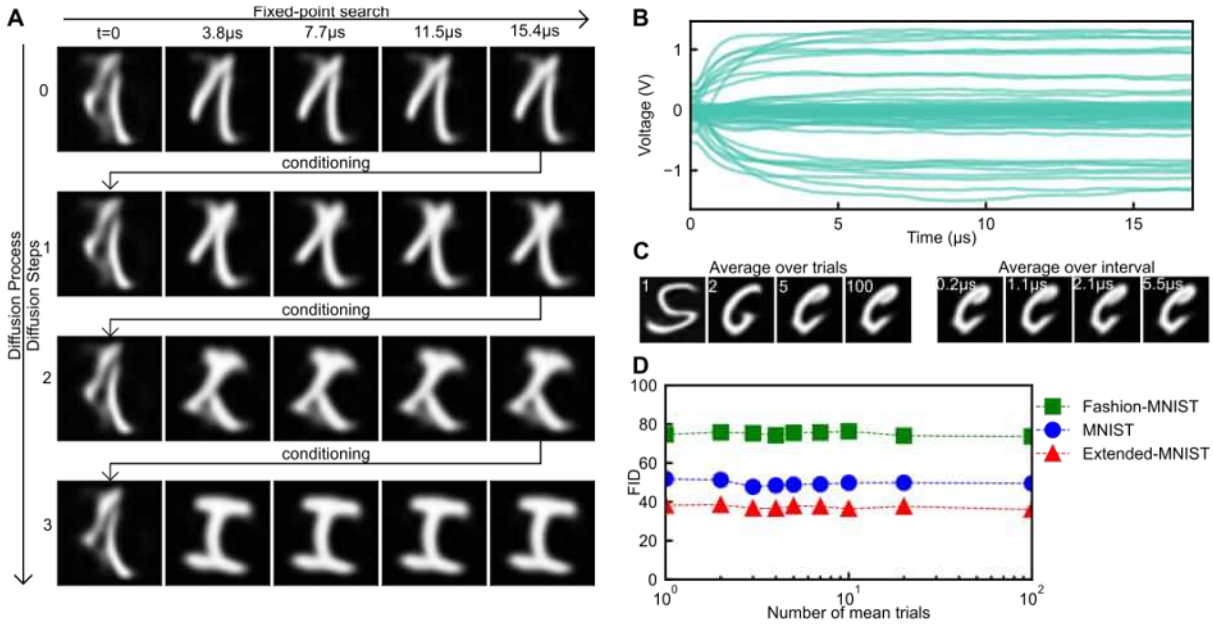


Fig. S8: Inference dynamics for latent ADM hardware experiments on EMNIST, chosen as a representative example among the considered datasets. **A,** Evolution of latent ADM inference for EMNIST. Each row shows the progression of the fixed-point search within a diffusion step, followed by successive diffusion steps (rows). **B,** Example traces of the 48 analog voltages over time for a representative fixed-point within a diffusion step. **C,** Effect of averaging across repeated experiments and within a fixed-point time window. **D,** FID as a function of the number of mean trials for MNIST, FMNIST and EMNIST, showing negligible improvement beyond five averages across datasets.

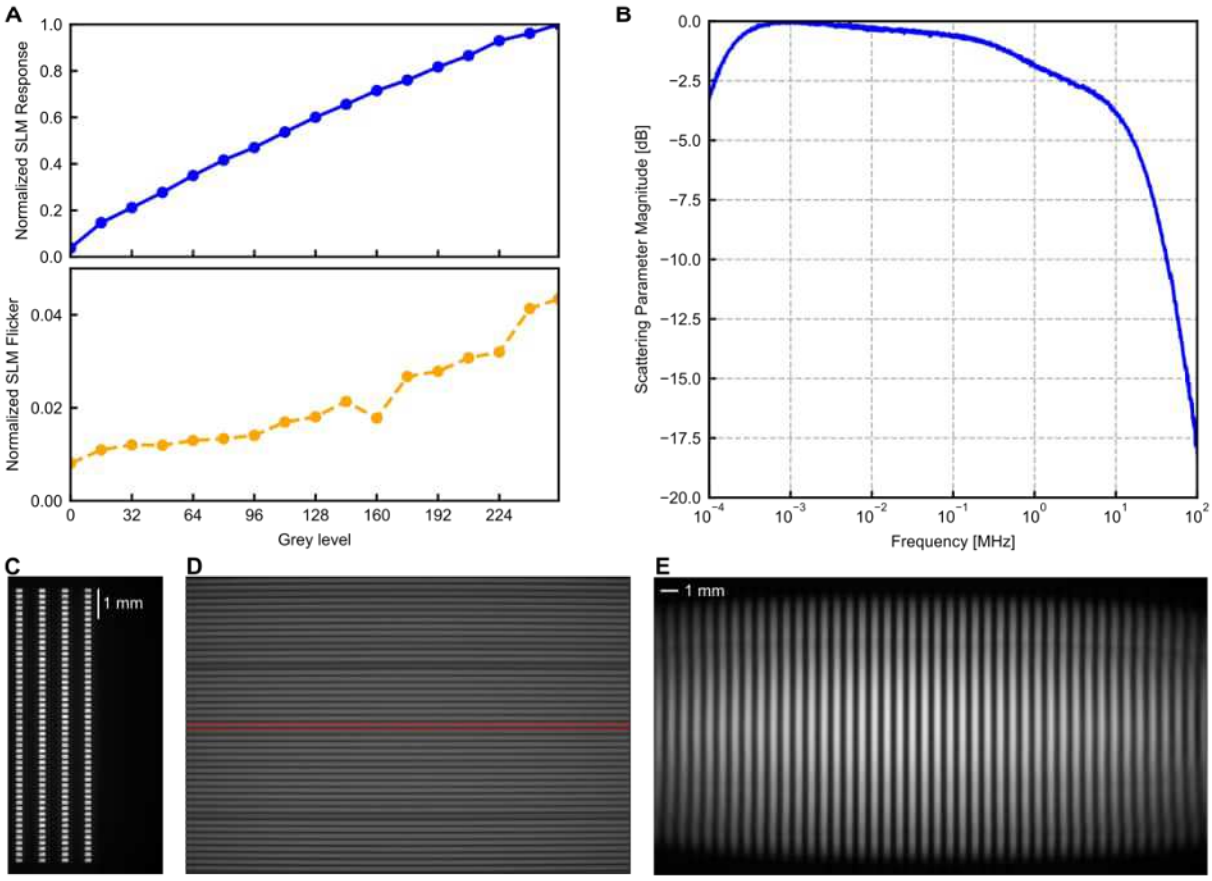


Fig. S9: Device and system characterization. **A,** Top: normalized static SLM response versus grey level. Bottom: normalized flicker amplitude measured at the matrix multiplier output; each distribution includes all the noise sources present in the opto-electronic system. Both SLMs present very comparable performances. **B,** Measured end-to-end bandwidth of the optical subsystem. Scattering parameter magnitude (dB) versus frequency for the LED–SLM–PD chain, measured using a Keysight E5061B network analyser. The micro-LEDs are driven at 10 mA, and the PD output was accessed via a breakout board. The -3 dB point occurs at 5 MHz, indicating the effective bandwidth of the current hardware. **C,** Camera image at the SLM plane showing the projected 4x48 LED array. The LEDs (25µm emitting height, 50µm pitch), are magnified by 3.86× and imaged onto the SLM plane, defining the optical input channels used for vector encoding. **D,** Camera at the same plane after inserting the merger lens, which optically combines the 4x48 LED array into 48 horizontal illumination stripes. When the SLM is placed at this plane, each stripe aligns with a row of programmable weights. The SLM has an 8.5µm pixel pitch and a 92% fill factor; each weight element is implemented using a block of 14x16 SLM pixels. **E,** Camera at the PD plane with vertical stripe patterns displayed on the SLM, illustrating how the merger performs optical summation across input channels.

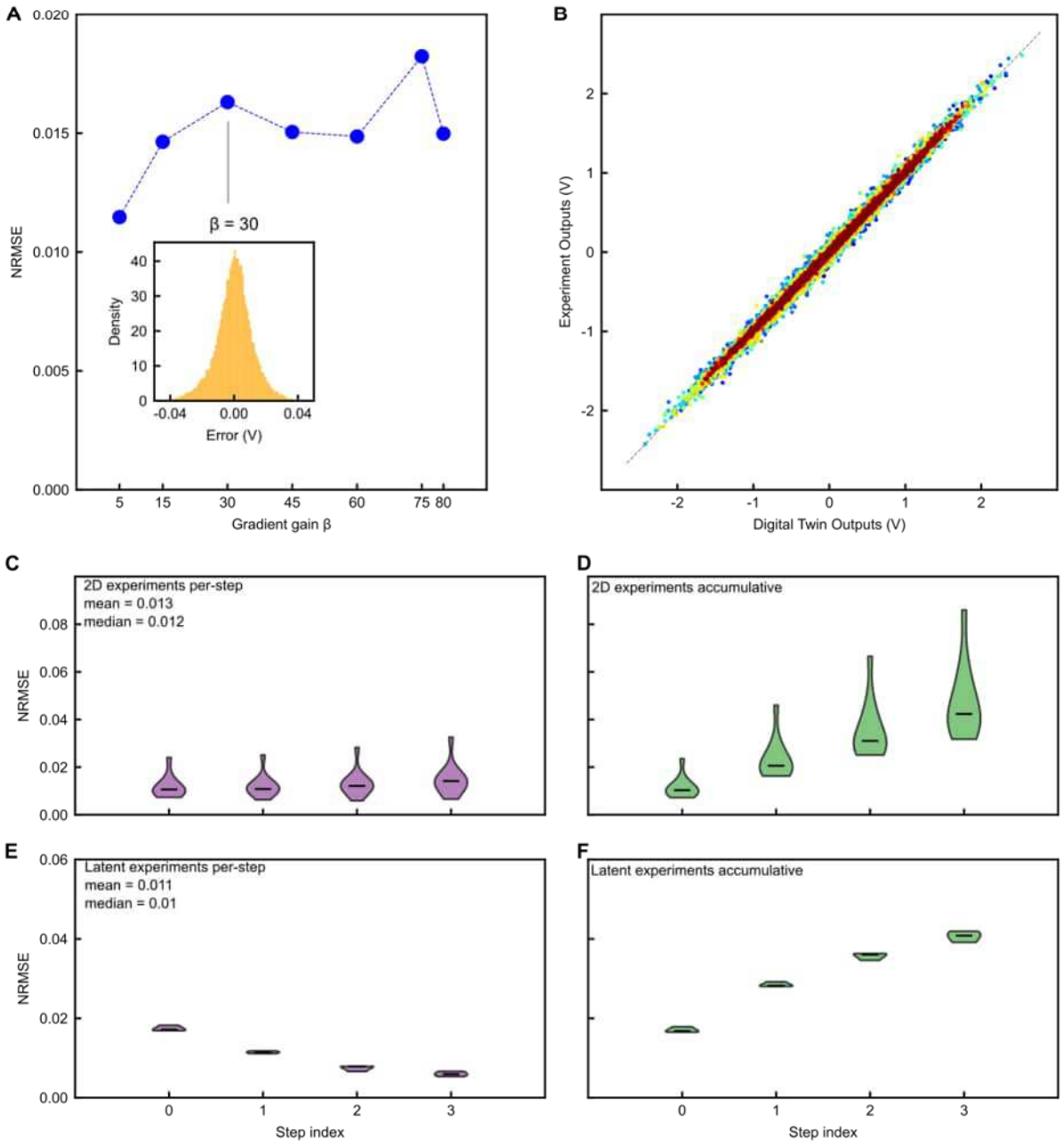


Fig. S10: Accuracy characterization of the analog optical hardware against digital twin. **A,** Open-loop accuracy of the matrix-multiplication path across gain settings β, reported as normalized root-mean-square error (NRMSE) averaged over 600 experiments per setting (100 random weight matrices × 6 input voltages). Inset shows the error distribution for $\beta = 30$. **B,** Closed-loop agreement between analog and digital-twin outputs over 1,000 experiments spanning random weight matrices and channel-wise gains α, β (mean NRMSE = 0.009). Colours demote different output channels. **C-F** Per-step and cumulative NRMSE during diffusion inference for 2D distributions (c, d) and latent-space tasks (e,f), aggregated across benchmarks. Horizontal bars indicate the median.

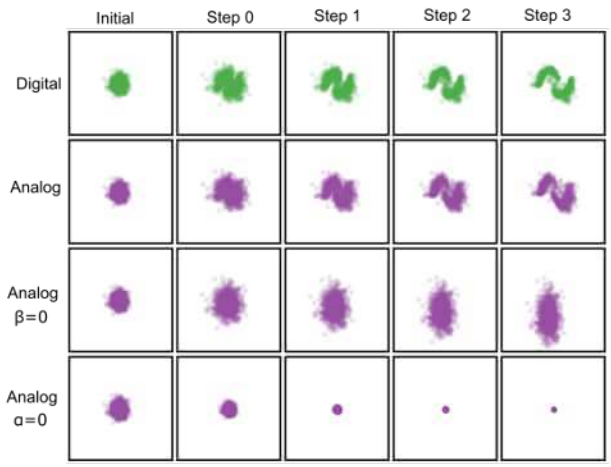


Fig. S11: Sensitivity of analog generation to hardware parameter settings to highlight the role of different parts of the system. Rows correspond to different execution modes: digital baseline, analog hardware with nominal parameter settings, analog hardware with optical gain disabled, i.e. optical subsystem off (β=0), and analog hardware with skip-connection disabled (α=0). Here, β denotes the optical amplification applied to the W$\sigma(z)$ term in (5), and α denotes the analog skip-connection fraction defined in (10). Columns show the evolution from the initial noise state through successive diffusion steps. Disabling either β or α prevents convergence to the target distribution, demonstrating that both optical and annealing dynamics are essential for successful analog diffusion inference.

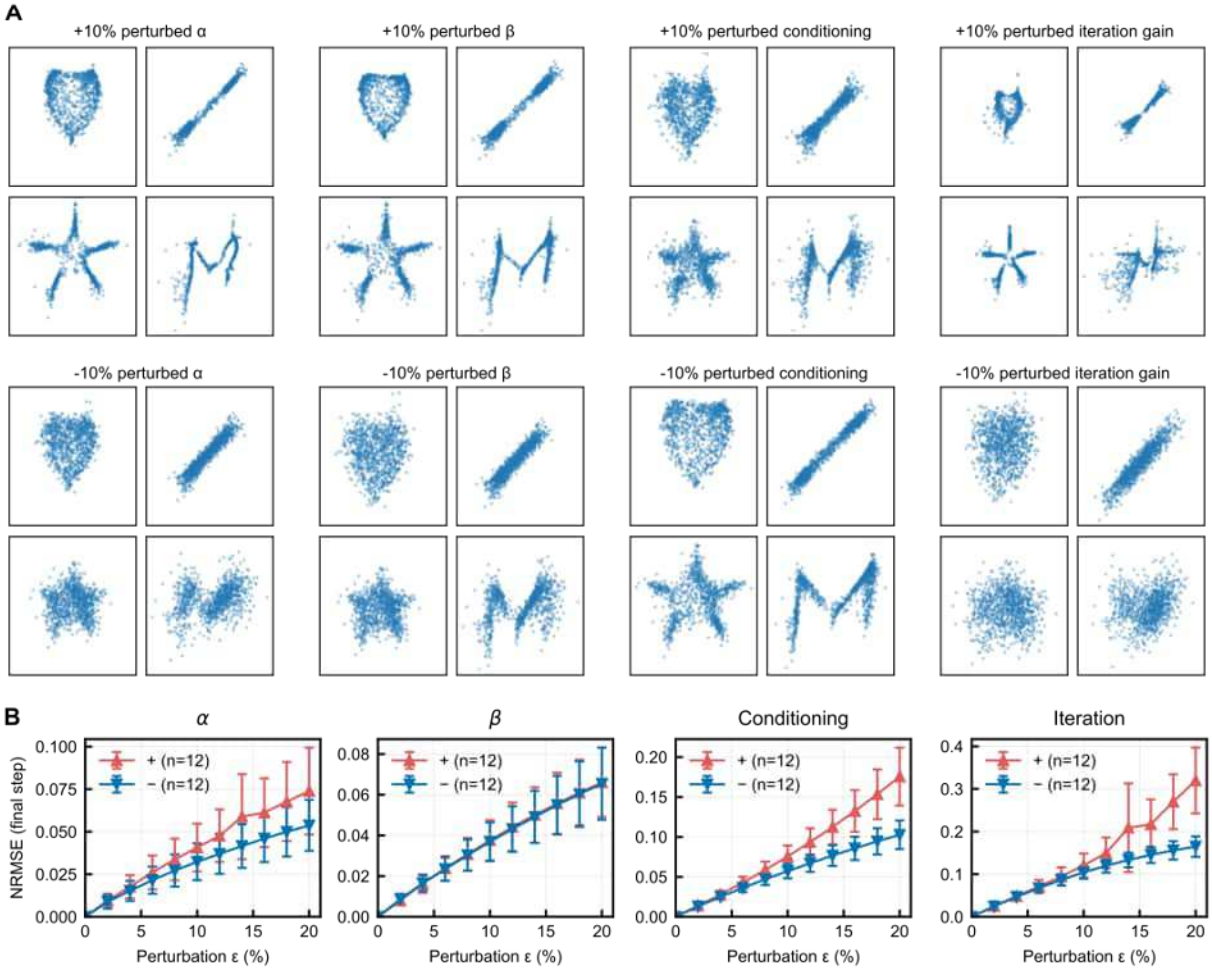


Fig. S12: Sensitivity of generation quality to multiplicative perturbations in hardware parameters. Each perturbation mode applies a uniform relative error ε element-wise across all 48 channels. (A) Representative final-step 2D distributions under ε=10% perturbation for four error sources (columns: skip connection gain α, gradient gain β, inter-step conditioning, and iteration gain). Top two rows: positive perturbation ($+\varepsilon$); bottom two rows: negative perturbation ($-\varepsilon$). (B) NRMSE of the final diffusion step as a function of ε, averaged across all 12 2D distributions shown in Fig. 2 (mean $\pm$ 1 s.d., $n = 12$). Red and blue curves denote positive and negative perturbation, respectively.

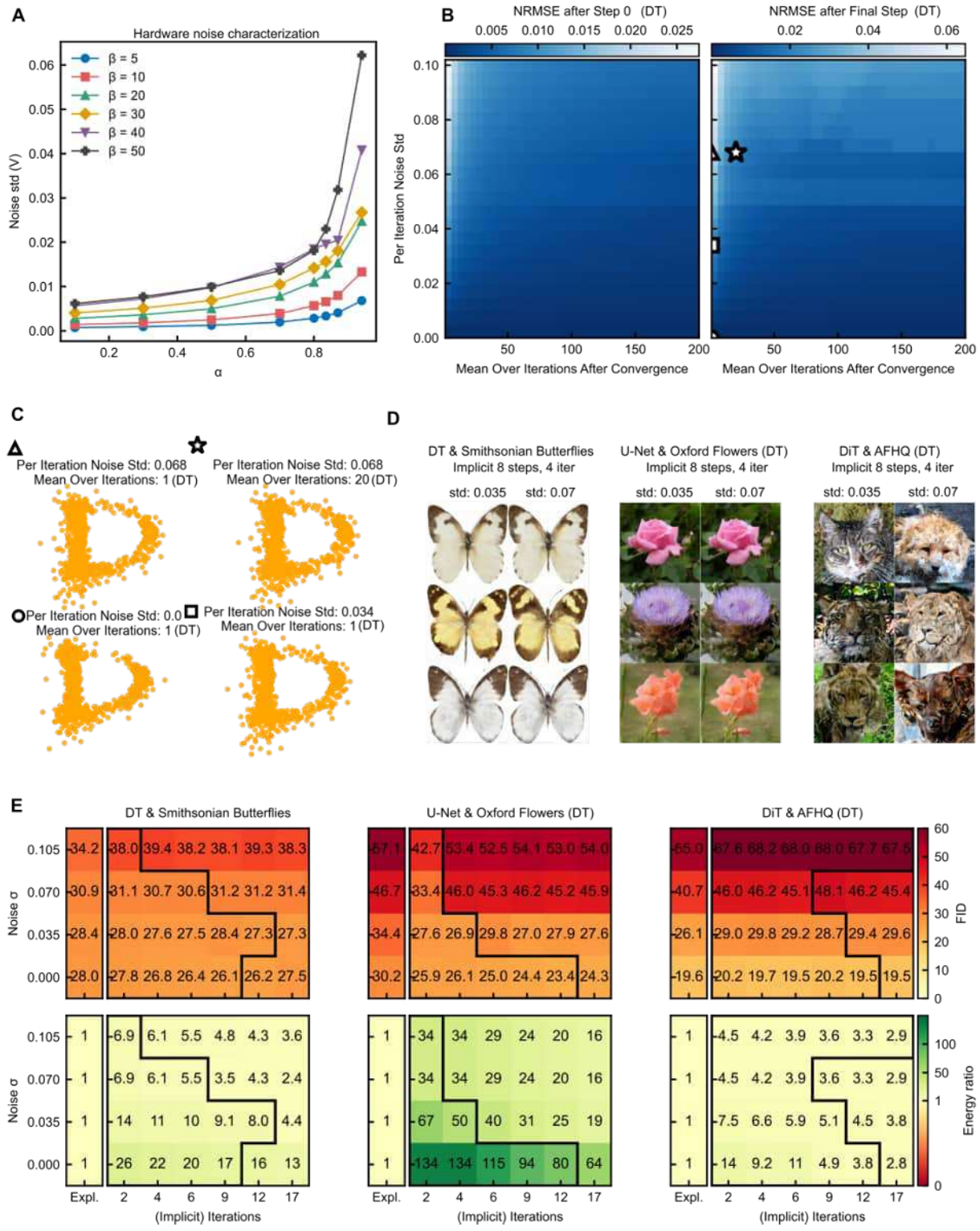


Fig. S13: Noise characterization. **A,** Measured hardware noise at various gain settings. **B,** Left: DT noise-tolerance characterization of implicit inference under per-iteration Gaussian noise using an examplar 2D distribution (letter D). Here we report the NRMSE at diffusion step 0 as a function of standard deviation and number of iterations averaged. The noise standard

-deviation range corresponds to levels observed on hardware under typical operating conditions. This quantifies intrinsic noise tolerance of fixed-point search and the benefit of in-loop averaging. Right: same analysis as in left panel, but reporting NRMSE at the final diffusion step after completing the full diffusion trajectory, capturing cumulative effect of per-iteration noise across per fixed-point search and across diffusion steps. **C,** Representative 2D distributions generated by the DT for two noise levels (standard deviation = 0 0.034, and 0.068) and two averaging settings (mean over iterations=1 and 20), illustrating that implicit diffusion inference preserves distributional structure under noise and that increased averaging improves sample fidelity. In analog hardware, this averaging incurs low marginal cost. **D,** Representative generated images at selected noise levels. **E,** FID and projected analog-hardware energy-efficiency gain under varying noise conditions. For reference, we assumed that analog hardware is 100× per-operation more efficient than digital, and a final digital decoding layer is considered, see supplementary section B.

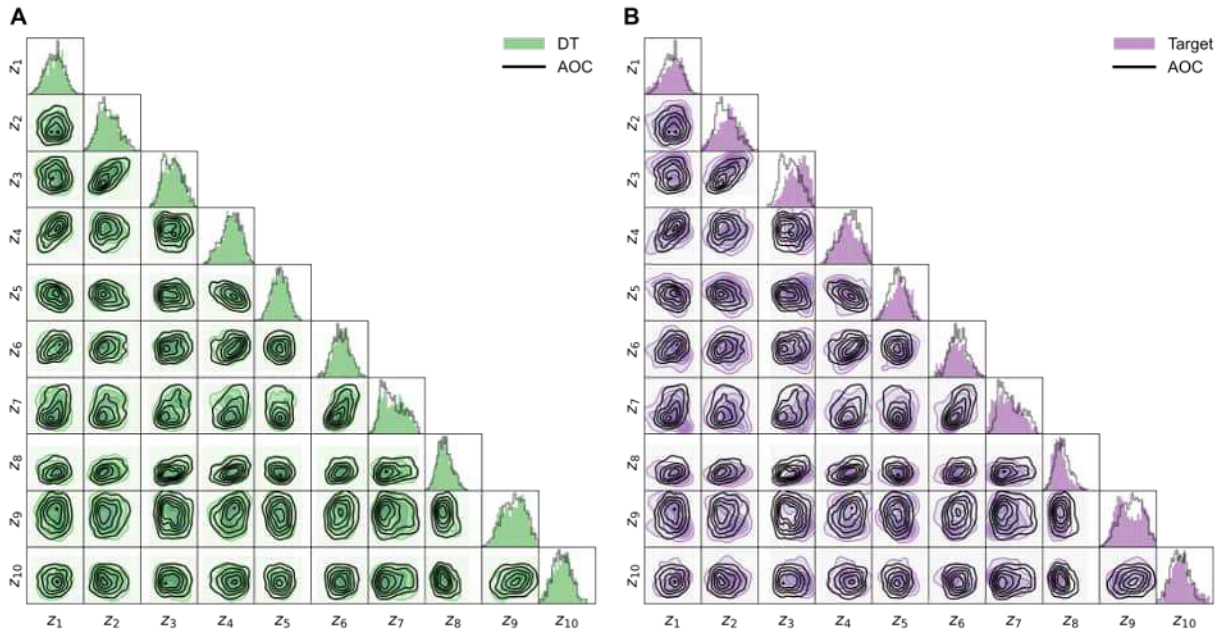


Fig. S14: Latent space distributions comparison for the MNIST hardware experiment, chosen as a representative example. **A,** Corner plot showing marginal (diagonal) and joint (lower triangle) distributions of latent variables z_1-z_{10} for the digital twin baseline (DT, green) and the AOC results (black contours). Diagonal panels show 1D histograms for each latent variable, while off-diagonal panels display kernel density contours representing pairwise joint distributions. For consistency, all off-diagonal axes ranges are limited to [-10, 10] for comparability. **B,** Same representation comparing AOC (black contours) with target distribution (purple).

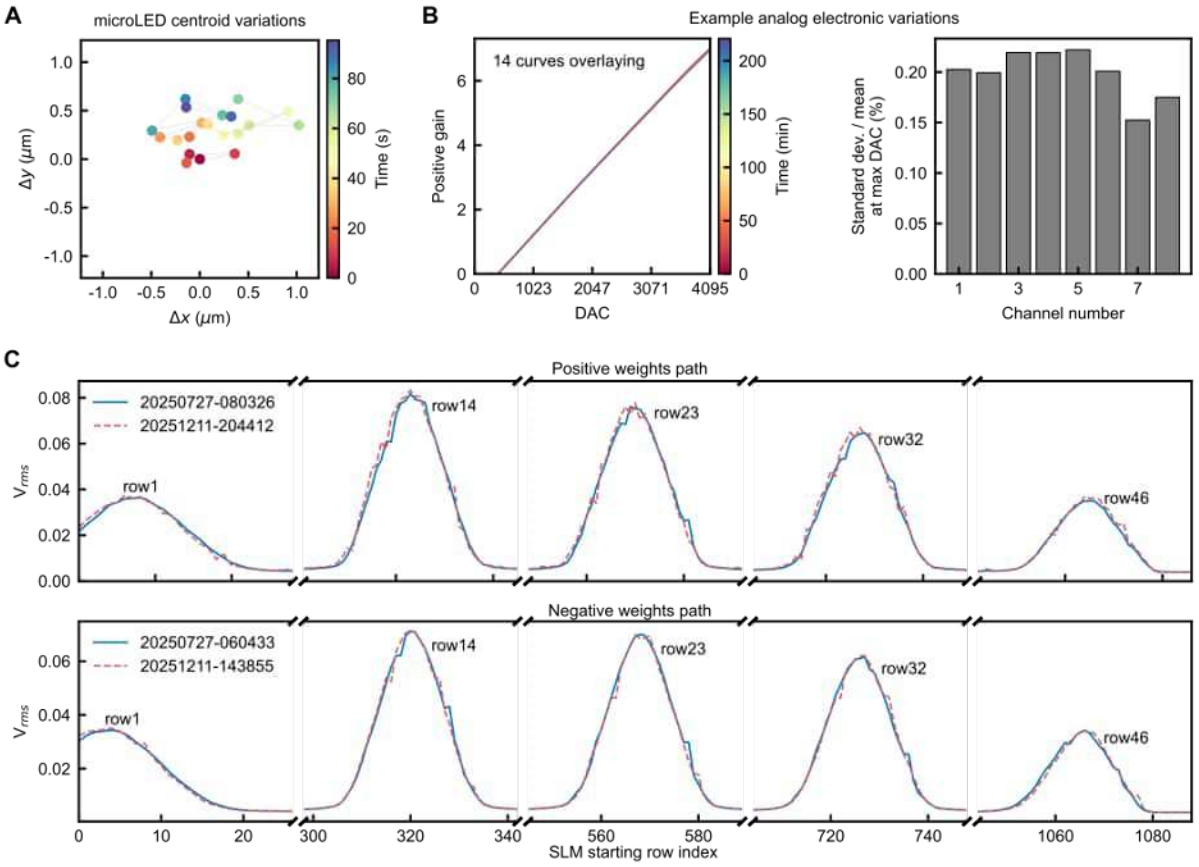


Fig. S15: Stability and reproducibility of representative analog hardware building blocks. **A,** MicroLED positional variation over time, measured by imaging the source onto a camera. **B,** An example of measured gain variation for the variable-gain amplifiers used to amplify the gradient signal of the positive path. **C,** Comparison of PD signals taken a season apart. A sine-wave voltage array is injected into the analog electronics board and propagates through the full signal chain, passing through the analog electronic nonlinearity (kept within the linear region of the tanh) and driving the microLEDs, whose light then traverses the imaging system to illuminate the SLMs. The SLMs display a row pattern, and the PD array measures the product of the input vector with this row-patterned matrix. We keep the row width fixed and sweep its starting index, so that plotting the RMS of the measured sine wave against the SLM starting row index turns the PD signal into a probe of the relative alignment between the SLMs and the LEDs, and the PDs and the SLMs. The top and bottom rows correspond to the positive- and negative-weights optical paths, and the two curves in each are labeled by their acquisition time. Over a season, the positional alignment typically shifts by less than ± 1 SLM pixel while its peak voltage stays unchanged. The ripple features visible on each curve corresponds to the SLM pixel pitch, as the imaging system images the discrete SLM pixel structure onto the PD plane.