

Rock Matrix Ultra-Fast Token

From a Constant-Size KV Cache to 1M Tokens/s per User

World-State EdgeRock DiffusionRock SiliconRock



Technical Summary

Uncompromised reasoning. Constant memory. Extreme single-stream speed.

AI is shifting from answering queries to synthesizing software on demand: given a task, a model generates the required agents, code, and toolchains on the fly. Latency is now the primary bottleneck. A functional agent routinely requires tens or hundreds of thousands of tokens across planning, code, tool interfaces, and tests. At today's typical 100–200 tokens/s per user, synthesis takes over ten minutes; at 1M tokens/s, it happens in real time. Making on-demand software viable requires a step-change in single-user decoding speed.

Memory is the first barrier. Mainstream models store context in a KV Cache that expands with every token, consuming more GPU memory, increasing per-token read bandwidth, reducing concurrency, and slowing down decoding as reasoning lengthens. Conventional optimizations compress per-token cache entries, which only delays the memory wall as long as the cache continues to grow. Rock Matrix takes a structural approach: instead of compressing the KV Cache, we eliminate its growth altogether.

At the core is the World-State: rather than accumulating an unbounded history, the model maintains a fixed-capacity state that continuously summarizes the task and environment as new information arrives. Bounded-memory architectures are not new, but prior designs typically lose distant details critical for multi-step reasoning. We solve this with Rock Memory Anchors (RMA). Like a climber setting anchors in rock, the model learns to identify high-value historical tokens and pin them in a fixed-size anchor bank, keeping them directly accessible long after they leave the local window. Powered by RMA, our first open-source model, EdgeRock-4B, cuts attention KV memory by 99.1% at 256K context while achieving 82.9% on AIME 2024–2026. Because per-request memory is constant, a single GPU supports far higher concurrency, boosting long-reasoning throughput to $6.42\times$ the source model and proportionally lowering cost per token.

Higher throughput cuts serving cost, yet a single user still waits on token-by-token autoregressive decoding. DiffusionRock changes the decoding paradigm: it drafts an entire token block in parallel and refines uncertain positions over a few iterations, updating the full block with each pass over the weights. A single request reaches 1,500 tokens/s, an order of magnitude faster than standard decoding, while maintaining a constant memory footprint.

Even so, GPU-based inference must still fetch model weights from off-chip memory on every forward pass; block decoding amortizes this transfer across tokens but cannot eliminate it. SiliconRock hardwires the backbone weights directly into silicon circuits, eliminating weight fetches from system memory or costly HBM, while storing the fixed-capacity World-State in on-chip SRAM. First, this slashes cost by removing HBM from the main path and simplifying packaging and interconnect. Second, it unlocks extreme speed: with zero weight movement and state co-located with compute, the entire generation loop executes inside a single die. Combining lower cost with ultra-low latency forms the foundation of Ultra-Fast Token: 1M tokens/s for a single user, enabling instant software synthesis, real-time world simulation, and direct robot control.

Three core technologies

Constant-capacity memory

Maintains task state while Rock Memory Anchors preserve critical history.

Block-parallel decoding

Drafts and refines full token blocks in parallel, cutting serial latency.

Zero off-chip weight reads

Backbone weights become circuits; state resides in on-chip SRAM.

Core claims

Capable: Uncompromised long-horizon reasoning.

Efficient: Lower serving cost per token.

Fast: Targeting 1M tokens/s per user.

Product lineup

EdgeRock

Constant memory; multimodal comprehension and reasoning; text output.

DiffusionRock

Block-parallel generation at 1,500 tokens/s per request, expanding toward any-to-any multimodal output.

SiliconRock

Full model hardwired on a single die with no HBM on the main path: lower cost and extreme speed.

Why Generation Must Get Faster

When software is synthesized on demand, decoding latency becomes the bottleneck.

Software used to be written ahead of time and delivered as static releases. Today, models can synthesize custom agents, code, and toolchains the moment a need arises, making bespoke, on-demand software practical for the first time. That shift rests on one condition: generation must finish within interactive latency, not leave the user waiting.

The decisive metric is single-user decoding speed: the rate at which a model generates tokens for an individual request. Beyond reading a prompt, the model must emit reasoning traces, executable code, and results, where every token adds to wait time. Agent workloads feel this most. Although the final answer shown to a person may span only a few hundred words, an agent plans, writes code, invokes tools, parses outputs, and self-corrects in a loop, routinely generating tens or hundreds of thousands of internal tokens per task. Most of those tokens are never read by a human; they are how the agent thinks and acts. Speed matters not because humans read faster, but because software and agents become usable sooner.

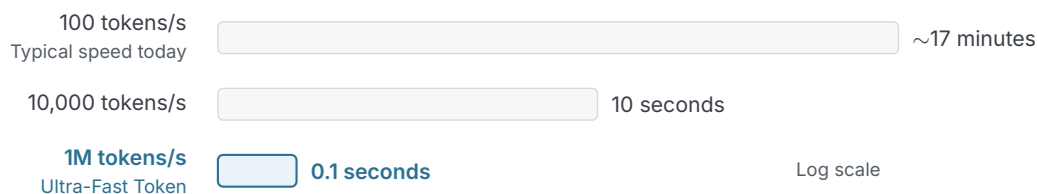


Figure 1. Generating 100,000 tokens requires ~17 minutes at 100 tokens/s, versus 0.1 seconds at 1M tokens/s.

Decoding speed also determines how continuously a model can operate. Live screen comprehension, field diagnostics, and environmental monitoring require uninterrupted state updates. Interactive digital worlds must respond within frame budgets, world models must roll out candidate trajectories under strict time constraints, and robots must adapt control actions as new observations arrive. Only when generation is fast enough can a general-purpose model remain inside real-time interaction and control loops.

High-speed models do not compete with frontier models; they complement them. Frontier models raise the ceiling of capability, while high-speed models bring mature capabilities into frequent, continuous deployment. Rock Matrix focuses on the latter: enabling strong models to sustain 1M tokens/s for a single user. We call this Ultra-Fast Token, and we are building toward it in three stages.

Rock Matrix Overview

Three physical walls cap single-user speed; our three stages remove each one.

The first is the **memory wall**. Standard models store history in a growing KV Cache, inflating both memory footprint and per-step read bandwidth as reasoning scales. Stage 1, **EdgeRock**, fixes the memory capacity: however long reasoning runs, memory stays flat and reasoning ability remains intact.

The second is the **sequential wall**. Autoregressive models decode one token per forward pass and cannot start the next until the last completes. Stage 2, **DiffusionRock**, drafts an entire block in parallel and refines it over a few iterations, slashing sequential steps while keeping memory flat.

The third is the **data-movement wall**. On a GPU, every decoding pass streams the full model weights from off-chip memory. Stage 3, **SiliconRock**, synthesizes weights into circuits and keeps the state in on-chip SRAM, eliminating weight movement and confining generation to a single die.

Each stage compounds the speedup toward 1M tokens/s per user. EdgeRock and DiffusionRock are standalone models that can be deployed independently or together.

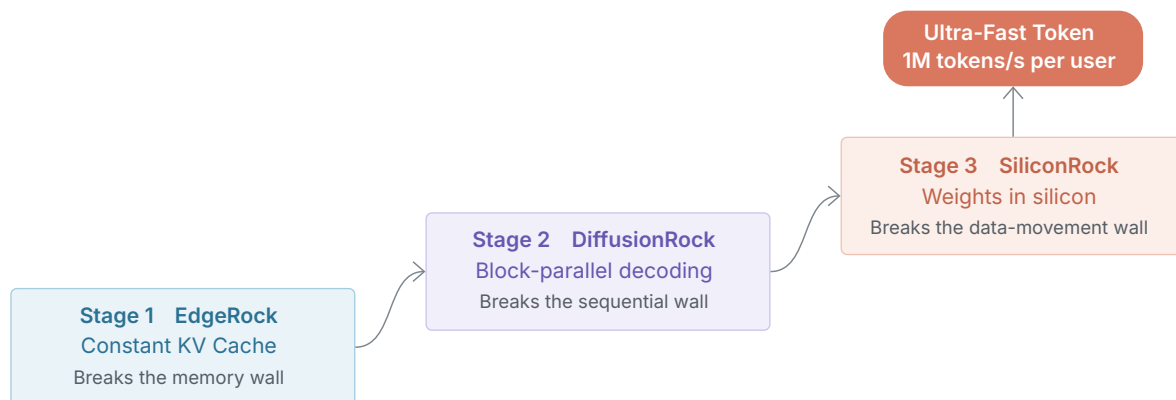


Figure 2. Three stages toward Ultra-Fast Token, each eliminating a fundamental bottleneck.

Highlights

- **Constant memory:** Fixed-size KV Cache uses zero extra GPU memory as reasoning scales.
- **Uncompromised reasoning:** Bounded state preserves frontier reasoning accuracy.
- **Cost efficiency:** Higher per-GPU concurrency drives down serving cost per token.
- **Order-of-magnitude speedup:** Single-user decoding speed increases by an order of magnitude.
- **Cloud and edge:** EdgeRock deploys across data centers and on edge devices.
- **Open source:** EdgeRock-4B model weights and inference code are publicly available.

Design Principles of Constant Memory

All three stages rest on a shared foundation: constant-capacity memory. It must satisfy three requirements at once: halt KV Cache growth, preserve critical long-range details, and convert memory savings directly into throughput and speed. Within this architecture, the World-State maintains an internal representation of the task and environment, while Rock Memory Anchors (RMA) govern selective long-term retention.

Principle 1: A Non-Growing KV Cache

Instead of compressing per-token cache entries, we bound total memory by design.

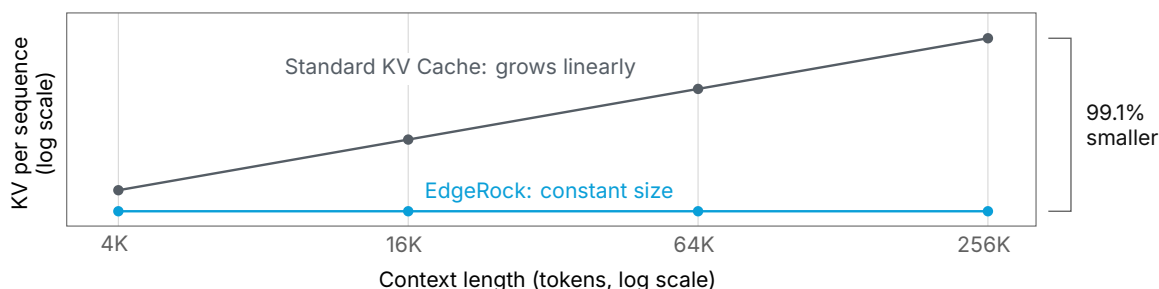


Figure 3. EdgeRock maintains a flat KV Cache as context scales, reducing KV memory by 99.1% at 256K tokens.

Prevailing approaches shrink or skip cached tokens: multi-head latent attention compresses each entry, sparse attention reads subsets of the cache, quantization lowers bit-width, and hybrid architectures such as Qwen3.5 and Kimi Linear retain full attention in only a quarter of their layers. While these techniques lower per-token overhead, the cache still grows linearly with sequence length—postponing the memory wall rather than removing it. Rock Matrix eliminates growth at the architectural level: history resides in a fixed-capacity state, so reasoning over 256K tokens requires the exact same memory footprint as 4K tokens. In production deployment, deterministic memory allocation matters as much as raw efficiency.

Principle 2: Uncompromised Reasoning

Selective retention preserves exact long-range details in bounded storage.

Bounded-memory models are not new, but prior designs either blur history into a lossy summary or discard tokens once they exit a local window. When fine-grained facts vanish, multi-step reasoning breaks down: across 16 bounded-state models we evaluated (up to 14B parameters), none exceeded 18.8% on AIME 2024–2026. The World-State partitions history into three stores: a recurrent state captures global context, a local window holds recent tokens, and an anchor bank preserves salient distant details. The first two build on Gated DeltaNet and sliding-window attention (SWA), while selective long-term retention is governed by Rock Memory Anchors (RMA). Every 256 tokens, RMA scores new candidates for importance, ranks them alongside existing anchors, and retains the top 1,024 entries per KV group. Retained anchors keep their original write scores and exact KV records, remaining directly readable long after leaving the local window. Every entry is also tagged with its source position, making retained memory directly inspectable. Retaining essential details and evicting redundancy keeps memory constant and reasoning sharp.

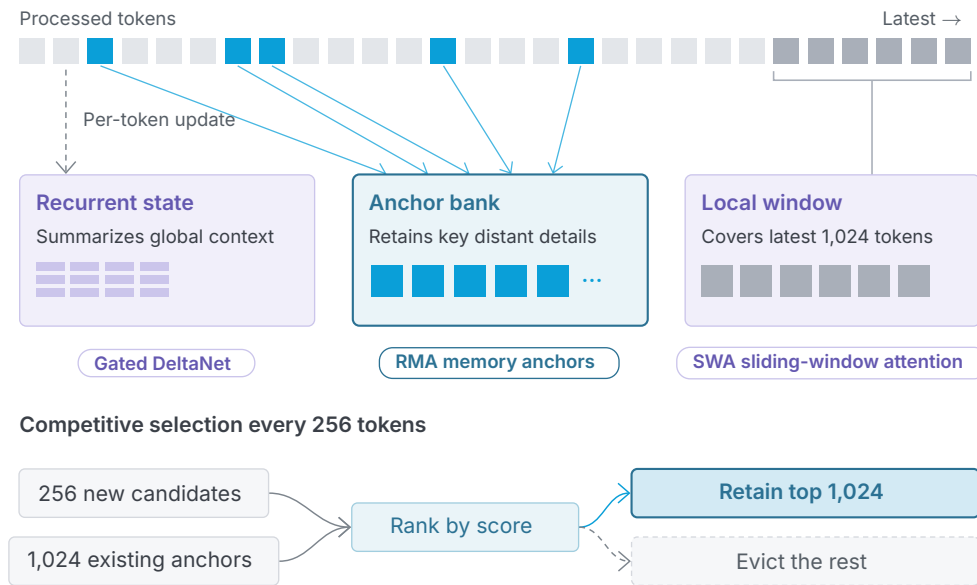


Figure 4. EdgeRock's three-tier constant memory: a recurrent state summarizes global context, a local window covers recent tokens, and the RMA anchor bank preserves key long-range details.

Principle 3: Co-Optimizing Memory and Compute

Saved memory becomes higher throughput on GPUs and lower latency in silicon.

On GPUs, capping per-request state size bounds both memory footprint and per-step read bandwidth, packing more concurrent sequences onto each device to multiply throughput and lower serving cost per token. On custom silicon, a deterministic state capacity allows on-chip SRAM to be provisioned to exact specifications right beside the compute units. Constant memory therefore optimizes memory utilization and data movement simultaneously.

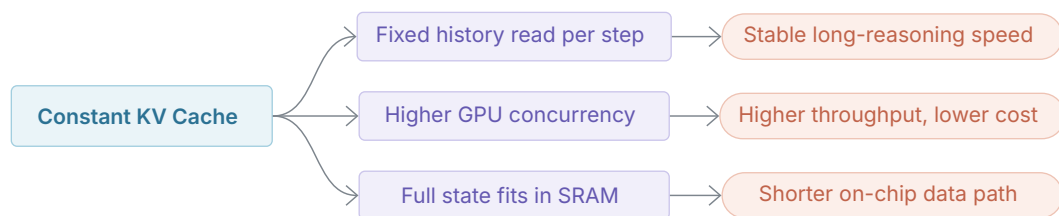


Figure 5. Constant memory yields dual dividends: higher throughput and lower cost on GPUs, plus shorter data paths and faster decoding on custom silicon.

EdgeRock: Strong Reasoning in Constant Memory

Multimodal comprehension and long-horizon reasoning in a fraction of the VRAM.

Built on the World-State architecture, EdgeRock is engineered for sustained comprehension and multi-step reasoning over long tasks. The open-source EdgeRock-4B is adapted from Qwen3.5-4B: it retains the 24 Gated DeltaNet recurrent layers, replaces the remaining 8 full-attention layers with 1,024-token sliding-window attention (SWA), and integrates Rock Memory Anchors. Training first warms up the anchor module for selective storage and retrieval, then jointly optimizes it with the backbone, and finally sharpens reasoning through supervised post-training and reinforcement learning with verifiable rewards. We are extending this conversion recipe across additional model families while pre-training native constant-memory models from scratch.

Constant memory requires no compromise on reasoning depth. EdgeRock-4B achieves 82.9% pass@1 on AIME 2024–2026, matching leading full-attention small models whose memory grows with sequence length, and outperforming the strongest bounded-state baseline in our evaluation by more than $4\times$. Memory anchors are the decisive factor: disabling RMA at inference time causes long-range retrieval accuracy in long contexts to collapse from 94.1% to 0.4%. Figure 6 benchmarks EdgeRock-4B across mathematics, general knowledge, scientific reasoning, and code generation.

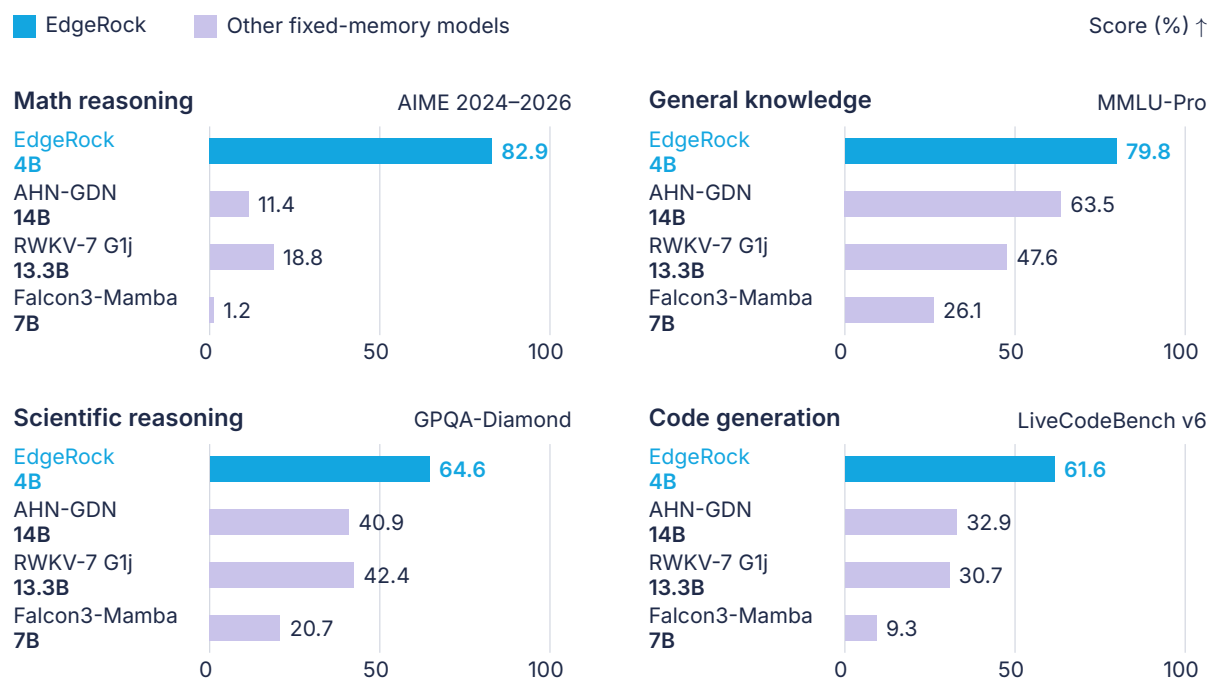


Figure 6. With only 4B parameters, EdgeRock-4B outperforms every larger 7B–14B fixed-memory model across all four benchmarks.

EdgeRock-4B accepts text and image inputs and produces text, code, and structured outputs, suiting multi-turn dialogue, long-document analysis, and long-horizon agent workflows. Extending comprehension to video, audio, and environmental signals comes next.

Performance: Longer Reasoning Without Higher Cost

Constant memory translates directly into serving efficiency. Consider GPU memory: when running 512 concurrent long-reasoning sequences on a single card (32K input and 32K output tokens each), EdgeRock-4B peaks at 61 GiB of VRAM, model weights included. Serving the same 512 sequences on the source model would require ~ 1 TiB for the KV Cache alone, the equivalent of 13 GPUs with 80 GB each. EdgeRock fits on one.

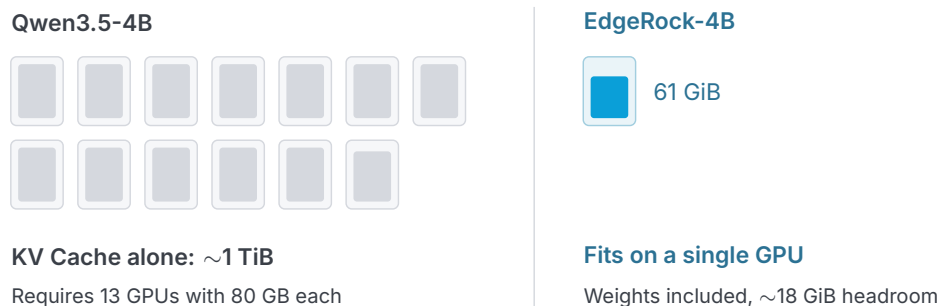


Figure 7. Across 512 concurrent long-reasoning sequences, the source model's KV Cache alone requires ~ 13 GPUs (80 GB each), whereas EdgeRock-4B fits on a single GPU with weights included.

Higher concurrency directly multiplies throughput. Generating several times more tokens per GPU-hour reduces the GPU cost per token to a fraction of the baseline—an advantage that widens as context length increases.

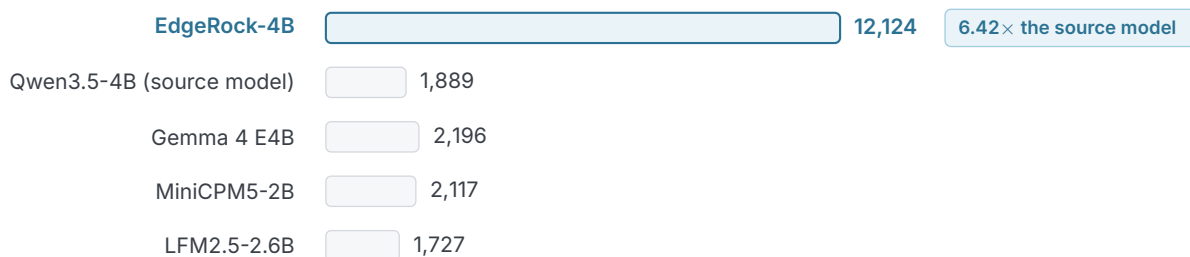


Figure 8. On a single GPU, EdgeRock-4B delivers $6.42\times$ the long-reasoning throughput of the source model (32K input, 32K output; tokens/s).

Throughput sets serving cost; single-user speed sets interactive wait time. EdgeRock also decodes a single request $1.54\times$ as fast as the source model. Cutting latency by another order of magnitude requires changing the decoding mechanism itself.

DiffusionRock: Fast Generation, Toward Any-to-Any

Parallel block decoding delivers 1,500 tokens/s on a single request.

Breaking the Sequential Bottleneck

Built on DiffusionGemma in its first release, DiffusionRock reaches 1,500 tokens/s for a single request, an order-of-magnitude speedup over the 100–200 tokens/s typical today.

The speedup comes from slashing off-chip weight reads. Mainstream models such as Qwen and DeepSeek are autoregressive: because each token depends on all preceding tokens, they decode strictly one token at a time. Generating 256 tokens requires streaming the full model weights from GPU memory 256 consecutive times. DiffusionRock replaces sequential decoding with block-parallel diffusion: it drafts all 256 tokens at once and refines uncertain positions over a few iterations. Each pass over the weights updates all 256 positions in parallel, putting otherwise idle GPU compute to work during single-request decoding. Memory remains flat throughout: DiffusionRock also manages context history with Rock Memory Anchors, so its KV Cache never grows with task length.

DiffusionRock is a standalone foundation model that generates answers, code, and tool parameters directly from prompts and context. What distinguishes it from EdgeRock is its decoding paradigm: EdgeRock decodes autoregressively for deep reasoning, whereas DiffusionRock decodes in parallel blocks for maximum speed.

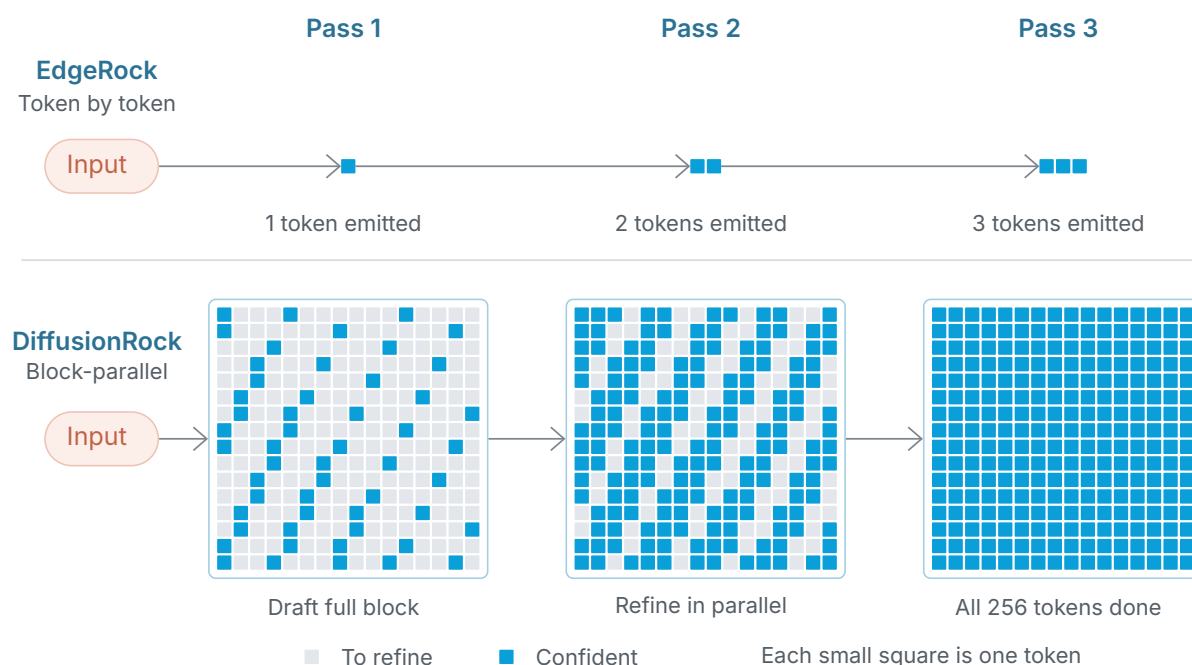


Figure 9. In three forward passes, autoregressive decoding emits 3 tokens, whereas DiffusionRock completes an entire 256-token block.

Toward an End-to-End World Model

Next, DiffusionRock will expand from text and code to images, audio, video, and actions, advancing toward any-to-any multimodal input and output. Evolving into a world model requires learning environmental dynamics and action consequences, enabling the model to synthesize future scenes and candidate actions for digital interaction, virtual environments, industrial simulation, and embodied AI.

Going a step further, we plan to unify comprehension and generation within a single model trained end to end. NVIDIA's Cosmos 3 world model for physical AI pairs two branches: an autoregressive understanding branch that interprets instructions and scenes to reason and plan, and a diffusion generation branch that synthesizes video, audio, and actions in blocks.

In our design, the understanding branch adopts EdgeRock and the generation branch adopts DiffusionRock, with RMA bounding memory across both. Each new observation updates memory, informs the next plan, and drives the next action.

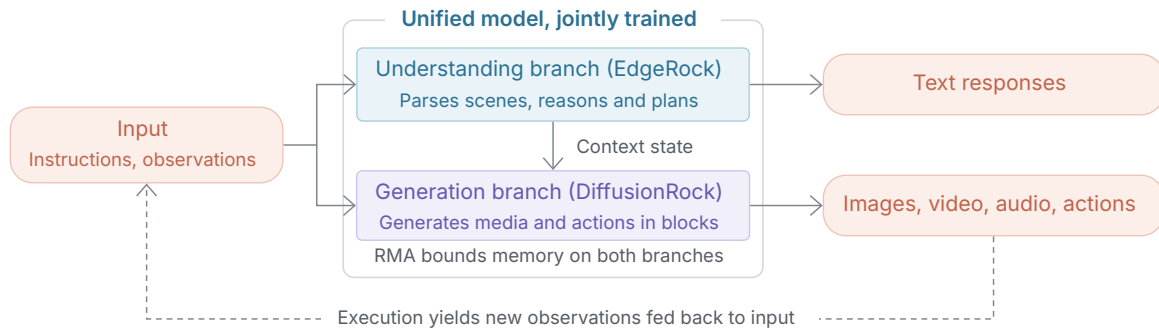


Figure 10. Applied to Cosmos 3: the understanding branch adopts EdgeRock, the generation branch adopts DiffusionRock, and memory remains constant on both sides.

Co-Designing the Model and the Chip

From first principles: static weights never move, dynamic state never grows.

Even with constant memory and block-parallel decoding, GPU inference hits a bandwidth wall: every forward pass still streams the model’s weights from off-chip memory. Consider a 2.1B-parameter model with 4-bit weights. Token-by-token decoding moves ~ 913 MB per step, so 1M tokens/s requires 913 TB/s of bandwidth. NVIDIA’s latest Rubin GPU delivers ~ 22 TB/s from HBM, more than $40\times$ short on paper: a gap no software optimization can close. Block decoding amortizes weight transfers across multiple tokens, but cannot eliminate them.

Weights that remain static during inference should not be fetched every pass, and state that updates every step should reside beside the compute units. Achieving both takes more than a new chip. Taalas HC1 hardwires weights onto a single HBM-free die and stores the KV Cache in on-chip SRAM, running Llama 3.1 8B at $\sim 17,000$ tokens/s per user. Yet Llama’s KV Cache grows with context, consuming more SRAM and increasing read traffic at longer contexts. Conversely, modifying only the model stops KV growth but leaves weight movement intact. Only co-designing the model and the chip eliminates both sources of data movement.

This co-design reduces cost and latency together. It is cheaper because removing HBM from the main path eliminates the most expensive component in modern accelerators, simplifies packaging and interconnect, and streamlines the system. It is faster because generation never waits on off-chip weight fetches, executing entirely within a single die.

Co-design does not mean starting from scratch. We replace only the unbounded attention cache while preserving the language, reasoning, and multimodal capabilities of mature models, allowing open-source advances to carry over directly.

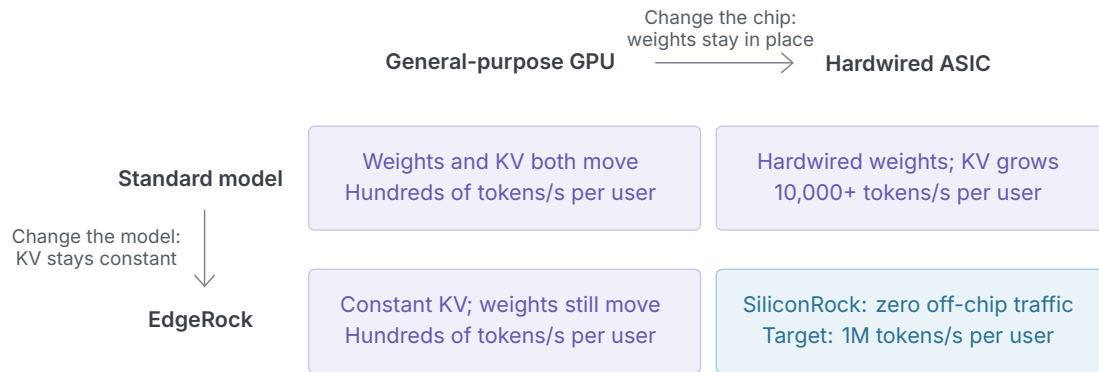


Figure 11. Modifying either the model or the chip removes only half the bottleneck; co-designing both eliminates off-chip data movement entirely.

SiliconRock: The Whole Model on One Chip

Every token traverses the full model on a single die within one microsecond.

SiliconRock compiles a converged model onto a single silicon die. Token ingestion, layer-by-layer computation, state updates, and next-token sampling all execute on chip; external I/O handles only requests and results. Both standalone models and jointly trained unified models follow this path toward 1M tokens/s per user. We welcome hardware and semiconductor partners to co-develop SiliconRock and bring models into silicon.

Weights as Circuits, State in Local SRAM

On a GPU, weights sit in off-chip memory and must be read in full for every token, leaving compute units waiting on data. SiliconRock inverts this by synthesizing each weight into the circuit datapath: as activations flow through, multiply-accumulate operations occur in place where the weight resides. Not a single byte of weights moves during generation: nothing is fetched from HBM or staged in SRAM.

SRAM is reserved for model state. Because the World-State has a fixed size that never scales with context length, it fits entirely in on-chip SRAM, partitioned layer by layer beside the compute units. For each token, every layer reads and updates its state locally, passing only the current token's intermediate activations to the next layer.

Next-token sampling also executes in place via hierarchical tree reduction. A 100K+ token vocabulary is partitioned across distributed groups; each group filters its top candidates locally, winners merge stage by stage, and the next token is sampled from just a few hundred finalists.

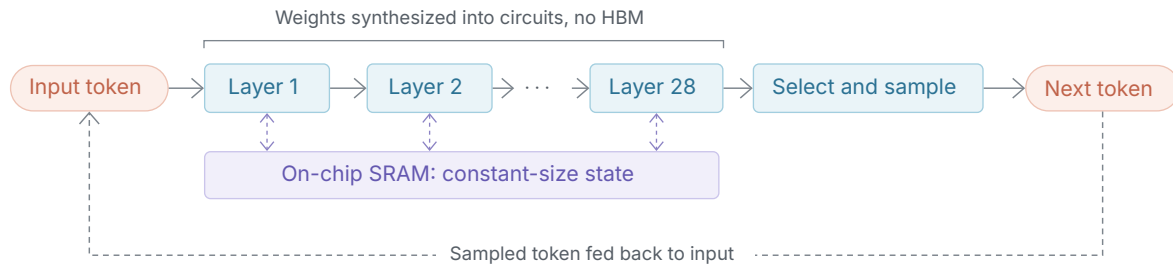


Figure 12. Each token completes the full on-chip path in under $1\ \mu\text{s}$: weights are embedded in circuits, and state resides in local SRAM.

One Token per Microsecond

Single-user decoding operates like a relay: one token must traverse all 28 layers, pass through sampling, and feed back to the input before the next token can start. Sustaining 1M tokens/s budgets just $1\ \mu\text{s}$ per token (1,000 clock cycles at 1 GHz). We scheduled our pipeline around this exact budget. Each layer gets 30 cycles, overlapping multiply-accumulates with local state updates; all 28 layers plus token selection, sampling, and feedback take 884 cycles—well under 1,000 cycles and equivalent to $\sim 1.13\text{M}$ tokens/s. Fitting into one microsecond is possible only because weights never move and state stays adjacent to compute.

Prototype Progress

We have built an end-to-end prototype on an early constant-memory model: all model weights are mapped into circuits, the physical design flow from model to chip layout runs end to end, and the circuits have passed functional correctness tests. This token-by-token prototype proves that constant memory and hardwired weights work together. Next, we will complete verification with commercial EDA tools and proceed to tape-out, followed by block-parallel scheduling for DiffusionRock to unite both speedups on one die.

Hardwired Backbone, Programmable Adaptation

Backbone weights and core compute structures are fixed in silicon, while LoRA adapters, domain vocabularies, robot action calibration, and safety thresholds reside in programmable storage for ongoing customization. While a model iterates rapidly, it runs on GPUs; once its capabilities stabilize, it is compiled into a SiliconRock chip.

Applications and Deployment

From on-demand software synthesis to world simulation and direct robot control.

For developers, integration is drop-in: Rock Matrix models are served through standard OpenAI-compatible APIs, so existing prompts, tools, and agent workflows run without modification. Applications expand outward one modality at a time: from text and code, to interactive video and audio, to robotic actions.

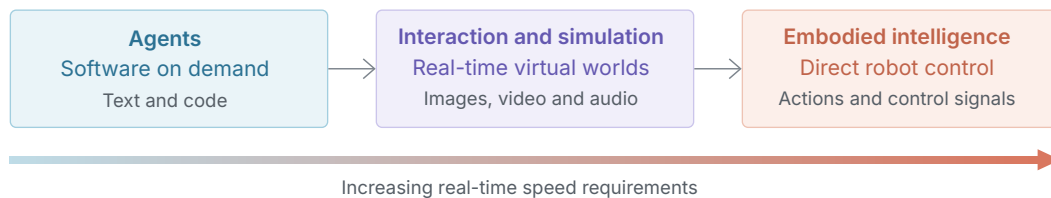


Figure 13. From synthesizing software to simulating worlds and controlling robots, each domain imposes stricter real-time latency demands.

Agents: Collaborative Acceleration with Frontier Models

In complex agent tasks, a frontier model handles planning and verification, while Rock Matrix fast models do the token-heavy work: generating code, tool parameters, and agent routines before returning results for verification. This division of labor is measured by end-to-end completion time and cost.

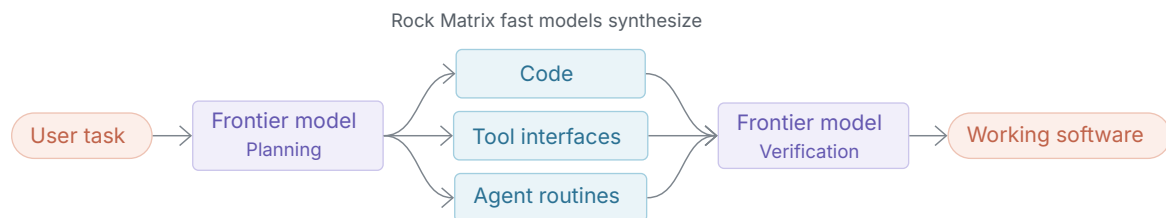


Figure 14. On-demand software synthesis: a frontier model plans and verifies, while Rock Matrix fast models generate the bulk of code, tool interfaces, and agent routines.

Interaction and Simulation

A world model is a learned simulator: given the current scene and an action, it predicts the next scene. World models such as Genie 3 and Cosmos 3 already generate interactive virtual worlds in real time to train robots and autonomous vehicles, yet remain constrained by two bottlenecks: per-frame latency and memory that grows with interaction length.

Rock Matrix resolves both. Next, we will bring the World-State to interactive environments so memory stays flat across long horizons, while using DiffusionRock for block-parallel generation and SiliconRock for on-chip execution so each frame arrives faster. Robots can then test actions, observe outcomes, and refine policies in simulation before real-world deployment.

Embodied Intelligence: Direct Control by Large Models

Today’s robots typically pair a “brain” with a “cerebellum.” A large model serves as the brain to interpret tasks and plan, while low-level control signals stem from a compact policy network (the cerebellum), distilled from the large model like a student from a teacher. This small model exists only because control runs at high frequency and large models decode too slowly to keep up. Having seen far less data, though, the small policy is easily disrupted by novel scenes or unfamiliar objects, losing the large model’s generalization at the control layer.

Recent experiments show that letting a general large model control the robot directly works better: with no task-specific fine-tuning, GPT-6 Astra outperforms specialized robot policies.

The missing ingredient is speed. Once a large model matches the control frequency itself, distilling a cerebellum becomes unnecessary. A world model such as Cosmos 3 can drive the robot directly, merging brain and cerebellum so generalization flows straight into control. This is Ultra-Fast Token's core role in embodied AI: DiffusionRock generates action sequences in parallel and SiliconRock runs the full model on chip to match the control loop, while constant memory sustains long-horizon operation.



Figure 15. Brain and cerebellum become one: a single foundation model comprehends the task, generates actions, and continuously corrects course from execution feedback.

Roadmap

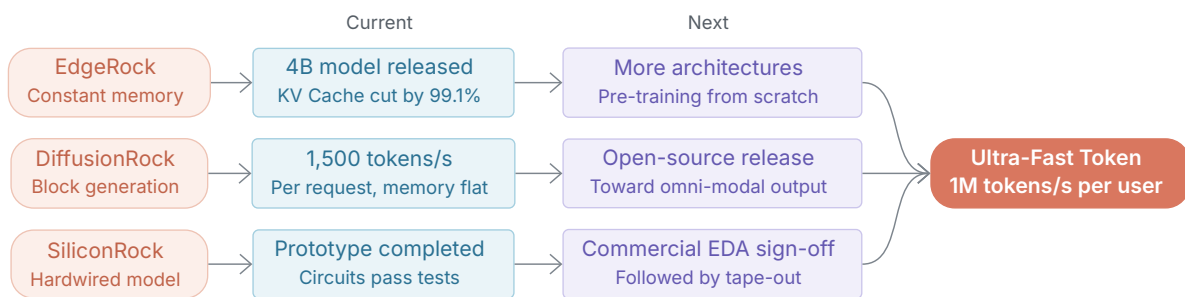


Figure 16. Three parallel tracks advancing toward Ultra-Fast Token.

Conclusion

From on-demand software synthesis to real-time world simulation and direct robot control, Rock Matrix advances along three tracks (constant memory, block-parallel generation, and hardwired silicon) toward 1M tokens/s for a single user, bringing the comprehension, reasoning, and actions of general models into real-time alignment with the physical world.

Models and code GitHub Hugging Face ModelScope Ollama

Technical report arXiv

Work with us High-speed token APIs for agents and inference workloads, private deployment, and joint model-and-chip co-design. Contact us at support@rockmatrix.ai.