Frozen Weights, Fast Silicon: Read-Only Memory as the Substrate for Frontier LLMs

Rewritten and expanded from the concept note "Using Fixed Logic Gates to Mimic Fast Read Only Memory to Augment Frontier LLMs" — second edition, adding the cartridge model. Michael Snyder (Pic2Mag), August 2026

Abstract. Deployed large language models read their weights but never write them, yet we store those immutable weights in the scarcest, most expensive memory technology we have (HBM/DRAM). This paper restates a simple proposal: burn a frontier model's frozen weights into cheap, fast read-only memory built with mature-fab diode-logic mask ROM, keep only a small "diff" in dynamic memory — and package that ROM the way the game industry always has, as changeable cartridges that plug straight into the video card. Web research shows the idea is no longer hypothetical: LoRA already is the diff mechanism, Apple already streams frozen weights from slow memory, and the startup Taalas (acquired by AMD in August 2026) already hardwires an LLM directly into silicon. This second edition adds the cartridge model, a survey of off-the-shelf plug-in bus standards, and a note on how known-distribution compression multiplies cartridge bandwidth.

1. The Observation: Trained Weights Are Read-Only

Once training finishes, a frontier model's weights — including its mixture-of-experts centers — never change again until a new version is trained. Everything the model knows, from the bronze age to its training cutoff, is frozen. Yet every inference server today stores that permanent content in dynamic RAM, paying DRAM and HBM prices, power, and packaging costs for data that is written exactly once. Viewed from the memory system, a deployed LLM is a **read-only publication**, and it deserves a read-only medium. The industry's "fast memory log jam" — years of HBM scarcity and pricing pressure — is partly self-inflicted: we are spending our best memory on our least dynamic data.

Once you realize that the weights and mixture-of-experts centers are permanent memory that never changes, the question stops being "how do we get more fast memory" and becomes "why is permanent data sitting in rewriteable memory at all?" The rest of this paper is that question, taken seriously.

2. The Encyclopedia Model: ROM Base + Live Diff

A print encyclopedia ships as twenty permanent volumes plus a small yearly update volume. The same split applies to models: ship the base weights as ROM "volumes," and ship a small diff file that lives in dynamic VRAM and patches the ROM image on the fly — re-issue the diff whenever the model is corrected or refreshed. This is not speculative: **LoRA already is the diff mechanism** [3]. Low-rank adaptation freezes the base weight matrix W and learns a tiny low-rank delta $\Delta W = BA$ (often <1% of the parameters), which is merged or applied at inference with no added latency. A ROM+diff architecture is therefore LoRA with the base matrix moved from DRAM into silicon: the diff in DRAM stays small precisely because the ROM base is allowed to be huge and cheap.

One more property of the analogy matters: **encyclopedia sets never go out of date**. A 1990 edition remains a correct record of what humanity knew in 1990. A model cartridge is the same kind of publication — everything from the bronze age to the training cutoff, frozen correctly — and the medium itself is archival. Mask ROM stores data as physical geometry, not trapped charge: flash fights leakage for a decade or two, DRAM needs constant refresh, but a mask pattern is stable for centuries. In the same way an encyclopedia set never expires, an LLM weight ROM cartridge is good for the next thousand years — and the small yearly update volume rides in DRAM.

3. The Cartridge Model: The New Solution Is the Old Solution

The consumer industry solved the "permanent content, generic machine" problem decades before LLMs. From the Atari 2600 (1977) through the Game Boy and the Nintendo 64, consoles shipped permanent content on swappable mask-ROM cartridges while the hardware stayed general-purpose: the console was the platform, the cartridge was the publication. The new solution is the same as the old solution we used years ago. **Put a cartridge slot on the video card.** The GPU and its DRAM remain general-purpose, and the model ships as a cartridge you can hold in your hand — swap the cartridge, swap the model.

Figure 1. The concept: a 5060 Ti-class video card with a changeable top-edge ROM cartridge ("KIMI 3"). The GPU and its DRAM stay generic; the model is the cartridge.

The connector, notably, is not a research problem. Removable high-bandwidth buses have been standardized many times over: Thunderbolt 5 moves 80 Gbps bidirectionally (up to 120 Gbps in one direction) over an ordinary USB-C plug [9]; OCuLink (SFF-8611) breaks four lanes of PCI Express out to external devices; the enterprise SFF-8639 (U.2) connector has hot-swapped NVMe storage at full PCIe speed for a decade [10] — to say nothing of the PCI Express card edge itself. A cartridge interface is a choice among existing standards, not an invention.

4. Hardware Paths

Diode-logic mask ROM. A diode-matrix ROM is the oldest and simplest ROM: each stored bit is the presence or absence of a diode at a grid crossing. It uses very few transistors and is compatible with essentially every CMOS process of the last four decades, so production can run on mature, fully-depreciated fabs instead of competing with GPUs and HBM for leading-edge capacity. A key practical property of mask ROM is that the data pattern lives only in the upper metal/via layers: base wafers can be fabricated in advance and warehoused, then a new model release is committed with a small number of final masks — turnaround of weeks, not the months a full custom chip needs.

Photonic logic is the more glamorous alternative, but it is cutting-edge and would compete directly with fast-memory manufacturing for advanced capacity — the opposite of the goal.

Mixture-of-experts tie-in. MoE models activate only a few experts per token. Keep the hot experts (and the router) in DRAM and let cold experts live in ROM, swapped in on demand. Apple's "LLM in a flash" showed that activation patterns are predictable enough to make this "windowing" work even over slow flash [2]; ROM with far better latency makes it easier still.

5. Compression Multiplies the Bus

There is a free lunch hiding in the interface math. Huffman coding (1952) [11] — and its modern relatives, arithmetic coding and ANS — can compress any known set of values when the relative frequency and data type of the values are known ahead of time. A burned model is exactly that case: the complete weight tensor and its histogram are known at mask time, so a near-optimal prefix code can be computed offline and the tiny decode table burned into the cartridge controller alongside the data. Quantized weight distributions are sharply bell-shaped, not uniform — a 4-bit weight spends its sixteen symbols very unevenly — so lossless entropy coding squeezes the stream well below its nominal width. Every bit of that squeeze is free bandwidth: the effective throughput of the cartridge link becomes bus rate × compression ratio, at zero cost to model quality, because the coding is lossless. Compression does not just shrink the cartridge — it speeds the bus up. Combined with a 4-bit ROM base and residual precision in the DRAM diff (Section 7), a modest connector starts to look a lot less modest.

6. Related Work: Others Have Had This Idea

The proposal sits squarely in a line of work that has moved from papers to products:

• **Taalas (existence proof, 2026).** The startup Taalas hardwires model weights directly into a model-specific ASIC. Its HC1 chip bakes a quantized Llama 3.1 8B into TSMC 6nm silicon — notably a mature, low-cost node, exactly the economics this note argues for — and claims ~17,000 tokens/s; AMD acquired the company in August 2026 [4][5][6].

• **Apple, “LLM in a flash” (2023)** [2]: run models far larger than DRAM by keeping parameters in flash and streaming them on demand — the software analog of ROM+diff.

• **LoRA (2021)** [3]: frozen base + small delta is the industry’s standard update format, as argued above.

• **ROM compute-in-memory research.** Academic work already implements neural networks in ROM-based compute-in-memory macros (e.g., ROM-SRAM hybrid CiM for edge AI [7]; surveys in [8]) — the diode-ROM idea has a research pedigree, not just a product one.

7. Extensions Added in This Rewrite

• **Put the multiply in the ROM.** The bottleneck in any ROM scheme is moving weights across a board. ROM compute-in-memory (multiply where the bits live) removes the interconnect problem entirely [7][8].

• **OTP/eFlash middle ground.** One-time-programmable or embedded-flash variants trade some density for field-updatability — a “slow ROM” for deployments that cannot wait for a new mask set.

• **Editioned releases.** Sell ROM sets like encyclopedia editions (“Llama 5, 2027 edition”) with a trade-in program; the DRAM diff carries owners between editions. The cartridge form factor (Section 3) makes the trade-in literal.

• **Quantize the base, refine with the diff.** Store a 4-bit base in ROM (as Taalas does [4]) and let the DRAM diff carry residual precision where it matters — then let entropy coding (Section 5) multiply the cartridge bandwidth on top.

Figure 2. Summary graphic — the social-media version of this paper's argument.

References

[1] M. Snyder, "Using Fixed Logic Gates to Mimic Fast Read Only Memory to Augment Frontier LLMs," concept note (docx), Pic2Mag, 2026.

[2] K. Alizadeh et al. (Apple), "LLM in a flash: Efficient Large Language Model Inference with Limited Memory," arXiv:2312.11514, 2023. https://arxiv.org/abs/2312.11514

[3] E. J. Hu et al., "LoRA: Low-Rank Adaptation of Large Language Models," arXiv:2106.09685, 2021. https://arxiv.org/abs/2106.09685

[4] Wavect, "Taalas HC1 Review: Hardwired LLM ASIC," 2026. https://wavect.io/blog/taalas-hc1-llm-asic-review/

[5] ComputeLeap, "AMD Buys Taalas: The Chip That Bakes Weights Into Silicon," Aug. 2026. https://www.computeleap.com/blog/amd-buys-taalas-weights-in-silicon/

[6] ExplainX, "AMD Acquires Taalas: The Chip That Etches Model Weights Into Silicon," Aug. 2026. https://explainx.ai/blog/amd-taalas-acquisition-etched-silicon-chip-august-2026

[7] "ROM-SRAM Hybrid Compute-in-Memory for Edge AI," Research Square preprint. https://assets-eu.researchsquare.com/files/rs-7749677/v1/55cc6b41cf5df14523b8aa32.pdf

[8] L. M. de Souza et al., "Towards Efficient In-memory Computing Hardware for Quantized Neural Networks: State-of-the-Art, Open Challenges and Perspectives," arXiv:2307.03936, 2023. https://arxiv.org/abs/2307.03936

[9] Intel Corp., "Thunderbolt 5 Technology Brief — 80 Gbps bi-directional, up to 120 Gbps," 2023. https://www.intel.com/content/www/us/en/products/docs/io/thunderbolt/thunderbolt-5.html

[10] SNIA SFF Technology Affiliate, SFF-8639 ("U.2") and SFF-8611 (OCuLink) connector specifications. https://www.snia.org/sff/specifications

[11] D. A. Huffman, "A Method for the Construction of Minimum-Redundancy Codes," Proceedings of the IRE, vol. 40, no. 9, pp. 1098–1101, 1952.