A CPU inside an LLM: compiling exact arithmetic circuits into pretrained weights

22 November 2026 · 8 min read · blogs.iictp.org/cpu-inside-an-llm-zero-gradient-circuit-grafting

Short answer: “A CPU Inside an LLM”, an arXiv preprint dated 19 November 2026 by Tanmay Sah, presents closed-form “circuit-grafting” primitives that compile deterministic computation, such as multi-digit arithmetic, directly into a pretrained model’s weights with no training examples and no runtime hooks. The author reports 99.17% and 100.00% accuracy on 6-digit arithmetic for two small models, against 42.2% and 61.7% for LoRA fine-tuning. This is an arXiv preprint. It has not been peer reviewed, and its numbers are the authors’ own claims.

TL;DR The short version

- 'A CPU Inside an LLM', an arXiv preprint dated 19 November 2026, compiles exact arithmetic circuits directly into a pretrained model's weights with no training examples.
- The author reports up to 100% accuracy on 6-digit arithmetic, with normal text resuming afterwards 98% to 100% of the time.
- It is a single-author preprint read only at abstract level, but a reminder that reliable computation can be built into a model rather than delegated to a tool.

Read online, listen to it, or change the text size: https://blogs.iictp.org/cpu-inside-an-llm-zero-gradient-circuit-grafting

What problem does it address?

Language models are unreliable at exact multi-digit arithmetic unless they are fine-tuned or call an external interpreter. This paper asks whether exact logic can be written into the weights themselves, deterministically, while leaving the model’s natural-language ability intact.

How it works, in plain terms

- The paper builds the circuits from three analytical mechanisms, described in the abstract: a two-layer “Mellin-GLU quadrature” over bias-free gated feed-forward layers that handles unknown per-token normalisation scaling; cross-token scale cancellation using attention and recurrent-style shifts; and axis-aligned width extension of the model with norm compensation.
- An automated five-stage compiler synthesises signed addition, subtraction and multiplication circuits of 4 to 8 digits in constant depth, plus Boolean and parity circuits.
- No gradient step is used; the weight changes are derived in closed form.

Results reported by the author

- On 6-digit arithmetic (N = 1,200, BF16), IBM Granite 4.2 3B reached 99.17% and Alibaba Qwen 3.5 4B reached 100.00% ALU-prefix accuracy, with 100% when an FP32 sidecar is used.
- Resumption of normal text after the computation was 98% to 100%.
- A 12.8K-sample LoRA baseline reached 42.2% and 61.7%, with 0% inline continuation.
- The grafted weights are a tiny fraction of the model: at most 0.036% and 0.057% non-zeros, respectively.
- Boolean (N = 1,728) and 5-bit majority and parity circuits scored 100%, including audits on a 42-layer Gemma 4 E4B model.

Limitations and open questions

- Preprint by a single author; not peer reviewed.
- Much of the vocabulary needs the full paper. Only the abstract was read, so the method details are summarised, not verified.
- Results are on specific small models and fixed-width integer arithmetic.
- It is unclear how the approach combines with fine-tuning, quantisation beyond the INT8 and INT4 memory figures given, or models with different normalisation.

What it means for practitioners

The paper is a reminder that reliable computation can be made part of a model rather than delegated to a tool call. Whether that is better than a calculator tool depends on latency, auditability and maintenance. See what is a context window and LLM quantization explained for related background.

Frequently asked questions

- **Is it fine-tuning?** No. The abstract says zero training examples and zero gradients.
- **Does it hurt language ability?** The author reports 98% to 100% mid-sentence resumption and says natural-language competence is preserved.
- **Which models?** Granite 4.2 3B and Qwen 3.5 4B for the headline numbers.

Related reading

- LLM quantization explained
- What is an LLM context window?
- What is LLM hallucination?
- AI Practitioner

Sources and currency

Sources were read on 22 November 2026. Products, rules and standards change, so check the linked originals before you rely on any detail.

Related credential: AI Practitioner. Explore the exam facts and syllabus.

Sources and further reading

1. Sah T. A CPU Inside an LLM: Zero-Gradient Compilation of Exact Circuits into Pretrained Weights. arXiv:2611.26450. 19 November 2026. arxiv.org/abs/2611.26450.

Published by IICTP, an independent technical credential body, on 22 November 2026. The current version of this post is at https://blogs.iictp.org/cpu-inside-an-llm-zero-gradient-circuit-grafting. More guides: blogs.iictp.org.