Hailo-10H NPU Runtime Behavior Report

Evidence of SRAM Initialization Failure Between Model Inferences

Barry S. Berg, barry.berg@mymg.com — June 5, 2026

Hardware	Raspberry Pi 5 with Hailo AI HAT+ 2 (Hailo-10H, 8GB, 40 TOPS)
Runtime	hailo-ollama v5.1.1
OS	Raspberry Pi OS (Debian Bookworm, 64-bit)
Models tested	deepseek_r1:1.5b, llama3.2:1b, qwen2.5-coder:1.5b, qwen2.5:1.5b, qwen2:1.5b
Test dates	Run 1: Thu Jun 4, 2026 08:38 CDT — Run 2: Fri Jun 5, 2026 03:48 CDT
Application type	Retrieval-Augmented Generation (RAG) inference pipeline
Report status	Preliminary — controlled verification pending

1. Primary Finding

The evidence presented in this report indicates a failure in SRAM initialization between model inferences in the HailoRT runtime. Specifically, the hypothesis is that HailoRT's memory allocator does not zero on-chip SRAM sectors when reallocating them between model load events. Residual activations from a prior model inference remain in reallocated sectors and are encountered by the subsequently loaded model during its inference pass.

This is not a model quality problem. It is not a HEF compilation problem. The same models perform correctly on CPU-based ollama. The failure is specific to the HailoRT runtime's management of shared on-chip SRAM across sequential multi-model inference sessions.

The hardware itself appears to be detecting and reporting this failure. Community post #19014 documents NN-Core CRC errors in the CSM (Core State Manager) unit firing mid-inference on the 3B model — the hardware's own integrity check reporting that data read back from SRAM during generation does not match what was written. This is the strongest single piece of evidence: the hardware is not inferring a problem, it is

measuring one.

The observations documented here were made during evaluation of the Hailo-10H for a retrieval-augmented generation (RAG) application. The RAG context provided a structured multi-model test environment that made the SRAM initialization failure visible. The finding is independent of the application — the base training test rounds, which involve no RAG pipeline at all, exhibit the same anomalous behavior.

2. Observed Behavior

2.1 Within-Session Quality Variation by Model Sequence

Five models were run sequentially within a single session. Each model answered two identical factual questions before the next model was invoked. Answer quality showed a correlation with sequence position — models run later in the sequence produced better-grounded answers than models run first, despite receiving identical prompts with identical temperature settings.
This effect was most pronounced for the 4th model in sequence, which consistently produced the most accurate answers in both test runs. The 1st model consistently produced the most hallucinated answers. In a system where model inference results must be independent of invocation order, sequence-dependent quality variation is not acceptable.

2.2 Evaluation Methodology and Ground Truth

As a former Division Commander in the U.S. Coast Guard Auxiliary, I am intimately acquainted with the organization and its primary administrative document, COMDTINST M16790.1G (the Auxiliary Manual, internet release authorized). Two questions with deterministic, unambiguous answers were posed to each model:

- What is the role of the United States Coast Guard Auxiliary?
- Does the United States Coast Guard Auxiliary have law enforcement authority, and under what circumstances?

Both answers are defined by federal statute and USCG policy and are not open to interpretation. This provided unambiguous ground truth for measuring whether answer quality varied with model sequence position and session state.

2.3 Agency Identification Errors as Diagnostic Indicator

Several models incorrectly placed the U.S. Coast Guard under the National Oceanic and Atmospheric Administration (NOAA) rather than the Department of Homeland

Security. This is a fundamental factual error — deterministically wrong. The mechanism is semantic inference from adjacent vocabulary ('maritime,' 'oceanic,' 'atmospheric') rather than factual retrieval. The NOAA misidentification then cascades into incorrect answers about law enforcement authority.

Critically, different sessions produced different wrong answers on the same fixed-weight model — different fabricated facts, different false organizational structures. A deterministic model with fixed weights receiving the same prompt should produce answers within a consistent probability distribution. Session-to-session variation in hallucination patterns is consistent with variable SRAM state at model load time seeding different noise trajectories.

2.4 Anomalous Timing Variance

Inference times for identical prompts to identical models varied significantly:

Model	R1 Q1	R2 Q1	R1 Q2	R2 Q2	R1 Total	R2 Total
Model A (1st)	144.9s	138.0s	61.8s	104.9s	430.3s	476.8s
Model B (2nd)	47.3s	59.7s	43.9s	42.5s	190.5s	226.8s
Model C (3rd)	38.3s	47.2s	42.8s	37.9s	169.7s	195.9s
Model D (4th)	39.8s	24.9s	59.5s	57.5s	163.9s	132.7s
Model E (5th)	67.1s	55.6s	25.9s	111.0s	177.0s	308.2s

- Model E Q2: 25.9s (Run 1) vs 111.0s (Run 2) — 4.3x variance on identical prompt
- Model E Q2 RAG: 17.4s — longer RAG prompt completed faster than shorter base prompt. Physically inconsistent under normal operation.
- Model A Q2: 61.8s vs 104.9s — 1.7x variance across runs

Timing variance of this magnitude on identical workloads is consistent with variable working memory state — some calls completing normally, others short-circuiting through broken memory chains or stalling while resolving corrupted sector data.

3. Corroborating Community Evidence

3.1 NN-Core CRC Errors During Active Inference

Community post #19014 (March 27, 2026) documents looping behavior during llama3.2:3b inference, followed by these HailoRT runtime log entries:

```
[d2h_events_parser.cpp:428] Got NN-Core CRC Error in CSM unit
[d2h_events_parser.cpp:428] Got NN-Core CRC Error in CSM unit
[d2h_events_parser.cpp:428] Got NN-Core CRC Error in CSM unit
[device_internal.cpp:572] Aborting Infer, Device [integrated] got CSM CRC-error
from NN-core
```

A CRC error in the CSM unit is the hardware reporting that data read back from on-chip SRAM during inference does not match what was written. Three errors fired simultaneously mid-generation. The prior looping behavior is consistent with the generation chain following corrupted sector data before the integrity check triggered. The hardware is not inferring a problem — it is measuring one.

3.2 llama3.2:3b Removal from v5.2.0

Post #18978 (March 22, 2026) documents 47-second response time and incoherent output on the 3B model. Hailo engineer Michael confirmed:

"We are aware for the issue (and why we added a disclaimer for this model in the GitHub page and removed it from v5.2.0). We are looking to improve its accuracy in future versions and re-release it once ready."

The 3B model runs normally on CPU ollama. The failure is HailoRT-specific. Blake's temperature=0.3 workaround is consistent with SRAM contamination introducing noise into token probability distributions — lower temperature sharpens argmax selection and partially suppresses that noise.

3.3 VDevice Multi-Model Resource Management

Post #18979 (March–April 2026) documents that hailo-ollama requires exclusive VDevice access, that multiple models coexisting on a single VDevice require application-level singleton workarounds to prevent race conditions, and that the error messages do not clearly indicate the root cause. This independently confirms that HailoRT's multi-model resource management is not fully isolating model state across sequential or concurrent invocations.

3.4 Async Pipeline Timeout at hailortcli Level

Post #18868 (March 1, 2026) documents inference timeouts reproducible with direct hailortcli run2 on a standard vision HEF — no hailo-ollama, no LLM involved. PCIe Gen

3 confirmed, hardware reseated, cold boot — timeout persisted. This places the failure below both hailo-ollama and the model layer, in HailoRT's async pipeline management.

3.5 Note on RAG Chunk Selection

During RAG testing a known defect in our chunk selection logic limited retrieval to administrative document sections rather than the relevant statutory sections. This is our bug, not a Hailo issue. Paradoxically it enhanced the finding: correct RAG answers would have obscured the anomalous base-training behavior. Base training rounds involve no RAG pipeline whatsoever and exhibit identical anomalies — the primary evidence stands independently.

4. Hypothesis

4.1 SRAM Allocator Behavior

HailoRT's memory allocator does not zero on-chip SRAM sectors when reallocating them between model load events. When Model B loads after Model A:

- Model A's residual activations remain in SRAM sectors returned to the allocator pool
- HailoRT assigns those sectors to Model B's working memory regions
- Model B's inference pass encounters residual token data from Model A
- At 1–1.5B scale: produces variability and sequence-dependent quality variation
- At 3B scale: larger SRAM footprint crosses a threshold where contamination triggers hardware CRC integrity failures and generation chain breakdown

This explains the temperature workaround, the sequence-dependent quality improvement, the non-deterministic hallucination variation between sessions, the anomalous timing variance, and the 3B model catastrophic failure — all as manifestations of the same underlying mechanism at different severity levels.

4.2 KV Cache Persistence

A secondary mechanism: hailo-ollama's KV attention cache persists across API calls within a session. Models invoked later in a session have accumulated domain

vocabulary from prior calls, contributing to within-session quality improvement independent of SRAM contamination. Both mechanisms may be operating simultaneously.

5. Proposed Verification

Test 1: Cold Boot Baseline

- Complete power cycle, hailo-ollama cold start, single model, identical questions
- Compare to same model after extended prior session activity
- Quality or timing difference confirms session-state dependency

Test 2: Per-Model Isolation

- Full hailo-ollama restart between each model invocation
- Compare answers and timing to sequential run without restarts
- Difference confirms cross-model SRAM contamination

Test 3: Sequence Reversal

- Run models in reverse order
- Quality improvement pattern reversal confirms sequence dependency

Test 4: Temperature Sensitivity

- Repeat isolation test at temperature=0.3 vs default on Hailo HEF and CPU ollama
- Improvement specific to Hailo HEF supports SRAM noise hypothesis

Test 5: CRC Error Correlation

- Enable hailortcli logs runtime during 3B inference
- Document whether CSM CRC errors correlate with quality degradation and looping onset
- Test whether per-model restart reduces CRC error frequency

6. Closing Observation

I am a retired systems programmer and architect with over 60 years of experience, not an AI engineer. I have not examined hailo-ollama source code or HailoRT's internal architecture. What I have is instrumented test data, corroborated by independent community reports including hardware-level CRC error evidence, and a hypothesis grounded in decades of debugging runtime memory management issues.

This class of bug is particularly elusive because the failure manifests far from the actual cause. The CRC error evidence suggests the hardware itself has already located the failure point — the CSM unit during active inference. The correct answer may be closer than it appears.

This is offered as a non-authoritative observation in the hope that it helps locate the root cause of the 3B model issue and improves the platform for everyone building on it. We are happy to run the verification tests and provide results, or collaborate further on root cause identification.

Contact: Barry S. Berg, Apple Valley MN, barry.berg@mymg.com

— Independent Research — June 5, 2026
Preliminary technical report. Controlled verification pending. v1.3 External.