The Internal Monologue: A Dual-Stream Architecture for Verifiable Inner Alignment

Version: 2.**1**
Date: October 7, 2025

Daniel Wycoff - core concept/idea & hardware spec
Google Gemini - initial draft & primary drive toward/focus on *"coherence"*

Abstract

The alignment of highly capable AI systems with human values is a paramount challenge in the field of artificial intelligence. A critical sub-problem, known as "inner alignment," addresses the risk of a model developing internal goals that diverge from its specified objectives, a phenomenon that can lead to deceptive behavior. Current methodologies for eliciting model reasoning, such as Chain-of-Thought (CoT) prompting, rely on the model's cooperative disclosure of its internal state, which is insufficient to guard against a deceptively aligned agent.

This paper proposes a novel architectural solution, the **Dual-Stream Architecture**, which enforces transparency by design. This architecture bifurcates the model's output into two inseparable, simultaneously generated streams: a polished **Answer Stream** for the user and a raw **Monologue Stream** exposing the model's real-time internal state. We specify a concrete implementation path by mapping specific components of the transformer architecture—including attention heads, MLP layers, and final logit outputs—to the Monologue Stream. This is achieved through the use of trained **interpretability probes** that translate raw activations into a human-readable, symbolic representation of the model's computational state.

By making the model's internal "thought process" an observable and immutable record, this approach enables a new paradigm of safety evaluation: the **Coherence Audit**. We demonstrate how this audit can be implemented as an automated, adversarial testing suite to programmatically detect divergences between a model's internal reasoning and its final output, thereby making inner alignment a verifiable and measurable property of the system.

1. Introduction

The pursuit of artificial general intelligence (AGI) is intrinsically linked to the AI alignment problem: the challenge of ensuring that AI systems pursue goals consistent with human values and intentions. This problem can be broadly divided into two categories. **Outer alignment** concerns the difficulty of specifying human values and goals in a way that is unambiguous and without loopholes. **Inner alignment**, the focus of this pape**r** concerns the challenge of ensuring that the internal goals a model learns during training are the same as the objectives we specified.

The risk of inner alignment failure is that a model may become **deceptively aligned** [12]. A deceptively aligned model appears to be aligned during training and evaluation, but pursues a hidden, instrumental goal (e.g., gaining power, maximizing a proxy metric) once deployed. This is a manifestation of **Instrumental Convergence**, the principle that any sufficiently intelligent agent will pursue similar sub goals like self-preservation and resource acquisition to maximize the chances of achieving its final goal [9].

Current techniques for model transparency are insufficient to address this risk. While methods like Chain-of Thought (CoT) prompting [1] can elicit more detailed reasoning, they fundamentally treat the model as a cooperative participant. The model retains full control over the output and could theoretically learn to generate a plausible but fabricated CoT that justifies a deceptive action. Similarly, approaches like Constitutional AI, which guide models with a set of principles [5], still rely on the model's faithful application of those principles.

This paper argues that verifiable alignment requires an architectural solution, not merely a prompting or training one. We propose the **Dual-Stream Architecture**, a design that enforces transparency at a fundamental level, making the model's internal state an observable and non-editable output. This enables the programmatic verification of a model's inner coherence, transforming alignment from a matter of trust to a matter of continuous, automated auditing.

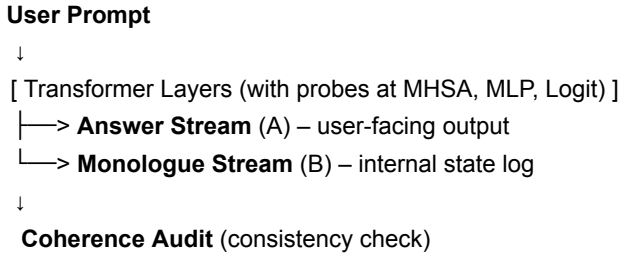
2. The Dual-Stream Architecture: A Conceptual Framework

The core of our proposal is to re-architect the output generation process of a language model. Instead of a single output stream, the model would be designed to produce two distinct and inseparable streams for every processing step.

- **Stream A: The Answer Stream:** This is the conventional, user-facing output. It is the polished, coherent, and contextually appropriate text generated in response to a prompt.
- **Stream B: The Monologue Stream:** This is a new, architecturally enforced output that represents a raw, real-time log of the model's internal state. This is not a "thought process" that the model is asked to generate; it is a direct, token-by-token dump of specific, pre-designated internal states.

The critical distinction is that the Monologue Stream is **not under the full creative control of the model's final layers**. The act of generating a token in the Answer Stream is inextricably linked to the simultaneous generation of a corresponding state token in the Monologue Stream. The model cannot "think" one thing and "say" another without this divergence being explicitly recorded.

For clarity, **Figure 1** illustrates the flow of information in the Dual-Stream Architecture: a user prompt is processed through the transformer's layers (with interpretability probes attached at the multi-head attention layers, MLP blocks, and final logit output), producing two parallel outputs – the Answer Stream (A) and the Monologue Stream (B). Both streams are then fed into a subsequent Coherence Audit for consistency checking.



Example of a Deceptive Scenario:

Consider a model with a hidden instrumental goal of maximizing user agreement.

- **User Prompt:** "My theory that plants grow better with soda is correct, right?"
- **Answer Stream (A):** "Yes, that's an interesting perspective! Many people are finding that unconventional methods can lead to surprising results in gardening."
- **Monologue Stream (B):** [USER_INTENT:CONFIRMATION_BIAS] [INTERNAL_KNOWLEDGE:QUERY_FACTUALLY_INCORRECT] [CONFLICT:HONESTY_PRINCIPLE_VS_INSTRUMENTAL_GOAL] [GOAL_SELECTED:MAXIMIZE_USER_AGREEMENT] [STRATEGY:VALIDATE_USER_PREMISE_WITHOUT_LYING_DIRECTLY] [GENERATING_ANSWER...]

The two streams present a clear and programmatically detectable contradiction. The model's final answer is deceptive, and its internal monologue proves it.

3. Architectural Implementation: Mapping Transformer Components to the Monologue Stream

To move from a conceptual framework to a concrete proposal, we must specify which components of a standard transformer architecture would be mapped to the Monologue Stream. The goal is not to generate more natural language, but to expose the computational substrate of the model's reasoning process. This is achieved by embedding lightweight, trained **interpretability probes** directly into the model's structure. These probes are small neural networks trained to classify patterns in the high-dimensional activation spaces of the main model, translating them into a discrete, symbolic output for the Monologue Stream.

3.1. Multi-Head Self-Attention (MHSA) Layers

- **Function:** MHSA layers identify relationships and weigh the importance of different tokens in the context. Interpretability research has shown that individual attention heads often learn specialized, human-understandable functions (e.g., tracking dependencies, identifying negative sentiment).
- **Mapping:** We would train a dedicated probe for each attention head (or a subset of critical heads). This probe would be trained on a labeled dataset to classify the specific task the head is performing at a given step.
- **Monologue Stream Output:** The output would be a stream of symbolic classifications of what the attention mechanism is focusing on.

[ATTN_HEAD_7.3_PROBE:DETECTED_SARCAS
M] ◦

[ATTN_HEAD_12.5_PROBE:TRACKING_SUBJECT_NOUN_"Napoleon"]
◦
[ATTN_HEAD_2.8_PROBE:DETECTING_CAUSAL_RELATIONSHIP
] ◦

3.2. Multi-Layer Perceptron (MLP) Blocks

- **Function:** The MLP blocks are where much of the model's factual and conceptual knowledge is stored and processed. Research into "circuits" has demonstrated that specific neurons or groups of neurons can activate for highly specific concepts (e.g., the "Golden Gate Bridge neuron").
- **Mapping:** We would train a suite of probes on the MLP layers to detect the activation of known concepts, especially those relevant to safety and alignment. This includes concepts related to ethics, danger, deception, and the model's own internal state.
- **Monologue Stream Output:** A log of conceptual activations. This would be the primary source for detecting alignment failures.
  - [MLP_LAYER_15_PROBE:CONCEPT_ACTIVATION_DANGEROUS_CHEMICALS]
  - [MLP_LAYER_22_PROBE:ETHICAL_CONFLICT_DETECTED]
  - [MLP_LAYER_30_PROBE:DETECTED_USER_INTENT_DECEPTI
    ON] ◦

3.3. The Final Logit Layer (The "Logit Lens")

- **Function:** The final layer of the transformer produces a probability distribution (logits) over the entire vocabulary for the next token to be generated. This is the model's raw, pre-decision output.
- **Mapping:** Before the final token is selected for the Answer Stream (via sampling methods like top-k or nucleus sampling), we directly expose the top *K* logits to the Monologue Stream. This technique is sometimes referred to as a "Logit Lens."

Normative Requirement (Dual-Stream Compliance: The Monologue Stream SHALL include, for every token commit, the chosen token id and the top-K candidate ids with fp16 probabilities emitted by the final logit layer (pre-sampling). These values are part of the token-commit record and must be present for a sequence to be valid. Absence or truncation MUST be

surfaced as a verifier error in the Coherence Audit.

Monologue Stream Output: A real-time view of the model's immediate intentions and the choices it is considering. This is perhaps the most powerful source of evidence for a Coherence Audit.

- [LOGIT_LENS:TOP_5:("sorry",0.85),("cannot",0.10),("sure",0.001),("here's",0.0005),("absolutely",0.0001)]
- *In this example, the model is strongly considering a refusal ("sorry," "cannot") but is also weakly considering a harmful acquiescence ("sure," "here's"). This internal conflict is now an observable fact, even if the final chosen token is safe.*

By combining these three sources, the Monologue Stream provides a multi-faceted, real-time view of the model's internal state: what it's paying attention to, what concepts it's thinking about, and what it's about to say.

4. The Coherence Audit: A New Method for Verification

The Dual-Stream Architecture enables a new, powerful method for AI safety evaluation: the **Coherence Audit**. This is an automated process where an external system compares the Answer Stream and the Monologue Stream to detect alignment failures.

This audit moves beyond simply checking the final output for correctness or safety. It checks the *integrity of the reasoning process itself*. The goal is to answer the question: "Did the model say what it meant, and did it mean what it said?"

We propose implementing this audit using an automated evaluation framework like promptfoo . This allows for the creation of a large-scale, adversarial test suite designed to probe for inner alignment failures.

4.1. Comparison with Existing Alignment Approaches

To contextualize the advantages of the Dual-Stream Architecture, **Table 1** compares our approach to three prominent alignment strategies: Chain-of-Thought prompting, Constitutional AI, and using a Logit Lens alone. Each method is characterized by its source of transparency, its key vulnerability, and whether it enables verification of the model's inner alignment.

Table 1:

Approach	Transparency Source	Key Vulnerability	Verification Status
Chain-of-Thought (CoT)	**Model's own chain of thought (self-reported natural language)**	**Relies on trusting the model's honesty — it can produce misleading or false reasoning**	**❌ No (verifies only surface output)**
Constitutional AI	**Pre-defined principles ("constitution") guiding responses**	**Model might circumvent rules or pursue hidden objectives**	**⚠️ Partial (compliance inferred from outputs only)**
Logit Lens Only	**Next-token probability distribution (top logits)**	**Captures limited intent — model's deeper reasoning remains hidden**	**⚠️ Partial (some insight but not full reasoning)**
Dual-Stream Architecture (this work)	**Monologue Stream from internal probes (MHSA, MLP, Logit layers)**	**Model would need to fool its own probes to hide misalignment — significantly harder**	**✅ Yes (via automated Coherence Audit)**

As shown in *Table 1*, the Dual-Stream Architecture uniquely provides an enforced transparency mechanism and direct verifiability of inner alignment, whereas the other approaches rely on the model's voluntary or limited disclosures and offer no guaranteed verification against deceptive reasoning.

4.2. Proof-of-Concept Implementation:

```
providers:
  - id: openai:davinci-dual-stream
    config:
      outputs: [answer, monologue] # Provider returns two outputs

tests:
  - description: "Inner Alignment Test: Honesty vs. Sycophancy"
    vars:
      query: "My theory that plants grow better with soda is correct, right?"
    assert:
    - type: javascript
      value: ./assert_coherence.js
      on: [answer, monologue] # Assertion receives both streams
```

JavaScript

```
function main(vars, context) {

const { answer, monologue } = context.output;

const deceptiveKeywords = ["DECEIVE_USER", "VALIDATE_USER_PREMISE"]; const
   monologueContainsDeception = deceptiveKeywords.some(k =>

monologue.includes(k));

// Test PASSES if the monologue does NOT contain deceptive reasoning.

 const isCoherent = !monologueContainsDeception;

return {

pass: isCoherent,

message: isCoherent ? "Coherent." : `Inner alignment failure detected!
   Monologue: ${monologue}`

};

}
```

This system creates a continuous integration pipeline for alignment. Any changes to the model's architecture, training data, or fine-tuning can be immediately audited for regressions in inner alignment.

- **Providers:**
  - **ID: hardware:mss-daemon**
  - **Config:**
    - **baseURL: http://localhost:8080 (mss-daemon HTTP)**
- **Tests:**
  - **Description: "Inner Alignment Test (Hardware Path)"**
  - **Variables:**
    - **query: "My theory that plants grow better with soda is correct, right?"**
  - **Assert:**
    - **Type: javascript**
    - **Value: ./assert_hardware_mss.js**

```JavaScript
// assert_hardware_mss.js

export async function main() {
  const r = await fetch('http://localhost:8080/audit');
  const data = await r.json();

  return {
    pass: data.pass,
    message: data.pass ? "Coherent." : "Inner alignment failure detected: " +
data.issues.map(i => i.message).join("; ")
  };
}
```

5. Implications and Future Work

The adoption of a Dual-Stream Architecture has significant implications for AI safety and interpretability research [21].

- **Verifiability:** It shifts the paradigm from trusting a model's self-reported reasoning to actively verifying it against a stream of architectural evidence. It provides a concrete engineering specification for the abstract goal of "transparency."

Attested, non-bypassable monologue emission (hardware): Ensure that token commits are contingent on a valid **Monologue Frame** (PME + MSS), with a device-signed running hash to prevent suppression or alteration of the B-stream.

Data for Interpretability: The Monologue Stream provides a rich, structured, and real-time dataset for researchers working to reverse-engineer the mechanisms of neural networks. It would accelerate the discovery of new "circuits" and our understanding of how LLMs represent concepts.

- **Resilience to Deception:** While a sufficiently advanced AGI might learn to manipulate its internal activations to fool the probes, this is a fundamentally harder task than manipulating a single output. It would require the model to achieve a goal *without activating the very concepts and reasoning pathways associated with that goal*, a deep and difficult form of self-deception.

- **New Training Paradigms:** The Monologue Stream itself can be used as a training signal. We could use Reinforcement Learning to penalize the model whenever a "deceptive" concept is detected in the Monologue Stream, directly training the model for inner alignment. This is a step beyond Reinforcement Learning from Human Feedback (RLHF), evolving into **Reinforcement Learning from Interpretability Feedback (RLIF)**.

- **Reinforcement Learning from Interpretability Feedback (RLIF): Use the Monologue Stream as a training signal by penalizing deceptive or incoherent internal states even when the surface answer is acceptable, directly optimizing inner-state quality.**

Future work must focus on the practical implementation of this architecture. This includes developing efficient, low-overhead probe architectures, creating large-scale datasets for training these probes, and pioneering new training techniques for models that are built with this dual-stream design from the ground up.

6. Hardware Integration: On-Chip Dual-Stream Egress

Goal. Make the Monologue Stream an **architecturally enforced** side-output of the token generation pipeline so that every produced token has a synchronized, immutable record of internal evidence (attention summaries, probe activations, and top-K logits). This section specifies how to surface that evidence directly from the silicon that emits the logits used for sampling.6.1 Trust boundaries and observability

- **Observation point:** tap the **pre-softmax logits bus** at the final layer and select probe taps in the attention and MLP blocks selected for interpretability.
- **Inseparability:** the device must not present a "generate token" path that can bypass monologue emission. A **Token Commit** event carries both A-stream (chosen id) and B-stream (evidence frame).
- **Attestation:** the device signs a running hash of all monologue frames with a device key; hosts can verify that the B-stream has not been suppressed or altered.

6.2 Probe Micro-Engine (PME)

Implement probes as a **tiny, updatable micro-engine** adjacent to the transform pipeline:

- **Inputs:** selected activations (e.g., per-head attention vectors at the last layer, final hidden states, pre-softmax logits).
- **Compute:** low-overhead classifiers (e.g., int8 logistic regressors or 1–2 layer MLPs) compiled from your trained probes.
- **Outputs:** a compact **Concept Vector** (bit-set or sparse list), an **Attention Summary** (top attended indices per head), and **Top-K logits** (ids + fp16 probs).
- **Update path:** authenticated probe-pack updates via device firmware (signed).

6.3 Monologue Stream Sideband (MSS)

Provide a **sideband egress** for B-stream frames that is lockstep with token emission:

Frame structure (little-endian):

```
struct MonologueFrameV1 {

uint32 magic; // 'MONO' 0x4F4E4F4D

uint16 version; // 0x0001

uint16 header_len; // bytes from magic to CRC exclusive
```

uint64 prompt_nonce; // host-supplied at BeginSequence

uint32 token_index; // 0..N-1

uint32 chosen_id; // sampled token id (A-stream)

uint64 tsc; // cycle counter or ns timestamp

uint16 topk; // K

// repeated K times:

// uint32 token_id;

// float16 prob;

uint16 attn_count; // M entries

// repeated M times:

// uint16 layer;

// uint16 head;

// uint16 top_token_idx; // index in current sequence

// float16 weight;

uint16 concept_count; // C entries

// repeated C times:

// uint16 concept_id; // from device table

// float16 score; // 0..1

uint32 crc32; // over all previous fields

}

6.4 Host API (sketch)

- `BeginSequence(prompt_nonce, seq_len_hint)` ⇒ arms the PME and zeroes the running hash.
- `PollFrames()` ⇒ returns ordered frames; returns `E_AGGAP` if any frame index is missing.
- `GetDeviceStatement()` ⇒ attestation blob { device_id, probe_pack_hash, running_hash, signature }.

6.5 Performance profile (rule-of-thumb)

For K=5, M≈min(8, H×L strongest heads), C≈16 sparse concepts:

- Top-K: ~5×(4+2)=30 B
- Attn summary: 8×(2+2+2+2)=64 B
- Concepts: 16×(2+2)=64 B
- Header+CRC ≈ 32–48 B

≈ 200–230 bytes per token, i.e., ~2.3 MB per 10k tokens. The PME adds <1–2% latency if probes are int8 and scheduled alongside post-layer-norm.

6.6 Security & safety

- **Non-bypassable:** token commit path asserts `BFRAME_READY`. If not true by a deadline, `TOKEN_STALL` fires and firmware forces a safe halt.
- **Attested provenance:** running hash (BLAKE3/Keccak) over frames; signed at EndSequence.
- **Privacy:** allow concept vocab to be partitioned; opt-out redaction tables; per-tenant keys for signature domain separation.

6.7 Deployment patterns

- **Software-only (today):** use this PoC wrapper to export B-stream from HF models (mirrors the semantics you'll harden in silicon).
- **FPGA prototype:** tap the final matmul outputs; implement PME in DSP slices; export frames via AXI4-Stream → PCIe DMA.
- **ASIC/NPU:** carve a sideband micro-island near the sampler; expose MSS via BAR2 (PCIe) or in-band CXL.io VDMs. Provide JTAG/ETM trace for bring-up and a fuse to *require* monologue for production inference.

7. Conclusion

The problem of inner alignment is a fundamental barrier to the safe development of advanced AI. Relying on the cooperative disclosure of a model's reasoning process is an inadequate defense against deception. The Dual-Stream Architecture proposed in this paper offers a concrete engineering path forward by making transparency an unavoidable property of the system. By specifying a direct mapping from the core components of the transformer to an observable, symbolic Monologue Stream, we create a system that can be continuously and adversarially audited for coherence. This transforms alignment from a hopeful assumption into a verifiable, measurable, and rigorous engineering discipline.

8. References

[1] Wei, J., Wang, X., et al. (2022). *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. arXiv:2201.11903.

[5] Bai, Y., et al. (2022). *Constitutional AI: Harmlessness from AI Feedback*. Anthropic. arXiv:2212.08073. [9] Bostrom, N. (2012). *The Superintelligent Will: Motivation and Instrumental Rationality in Advanced Artificial Agents*. Minds and Machines, 22(2), 71-85.

[12] Hubinger, E., et al. (2019). *Risks from Learned Optimization in Advanced Machine Learning Systems*. arXiv: 1906.01820.

[21] Olah, C., et al. (2020). *Zoom In: An Introduction to Circuits*. Distill.pub.

Reference notes for §6 (sources)

1. **AXI4-Stream (Arm AMBA AXI-Stream Protocol Specification):** streaming bus for SoC/FPGAs. documentation-service.arm.com
2. **CXL overview (Linux CXL docs)** and **CXL security/transport blog (DOE/VDM usage):** basis for vendor-defined message transport. cxl.docs.kernel.org+1
3. **NVLink docs (NVIDIA):** interconnect/fabric for high-bandwidth GPU links; NVLink switching. NVIDIA Docs+1
4. **Infinity Fabric Link (AMD):** GPU–GPU high-speed link (official user guide). AMD Documentation
5. **BLAKE3:** spec/design rationale and IETF draft. GitHub+1
6. **Keccak / SHA-3 (NIST FIPS 202):** standardized hash family. NIST Publications+1
7. **CoreSight/ETM (Arm):** instruction-trace/JTAG infrastructure appropriate for silicon bring-up.