

v3.4-STK-003 – Modular LLM Architecture & Token Logic

Document ID: v3.4-STK-003 | Status: Final | Version: v3.4

Date: 2025-03-22

Author: Take Back Your Data – Language Systems

Document Type: Public / Certification / Internal

1. Purpose & Scope

This document defines the modular design of the MaxOneOpen LLM and the token logic used for optimized, sovereign inference and minimal compute costs. It ensures that LLM components can be adapted, extended, and validated independently while preserving global system compliance.

2. Modular LLM Design

- Model architecture is divided into interchangeable modules:
 - Context Handler (pre-token logic)
 - Core Transformer Stack (LLM)
 - Control Adapter (external signal handling)
 - Postprocessor & Signature Block
- Modules can be optimized or replaced individually
- Specialization is achieved by creating purpose-specific forks with reduced token windows

Note: For detailed module-to-module interface specifications (data format, timing, payload), refer to Fork Blueprint §4.2.

3. Token Logic & Cost Minimization

- Token usage is measured via Token Work Unit (TWU): token + context + return
- Each twin has a budget envelope (in tokens/sec or tokens/task)
- Inference runtime applies token efficiency model:
 - Early-exit if context saturated
 - No token repetition / loop collapse
 - Prompt chaining compressed into logical kernels

4. Adaptability & Forking

- Teams can fork specific modules (e.g. control adapter) without retraining the full model
- Certified module hashes must be published and linked to the parent model
- Forked models must register a new scope and describe divergence from standard logic

Note: Implementation-specific variations and fork interfaces must comply with certified schemas (see Fork Blueprint §5.3).

5. Token Contract Interface (TCI)

TCI defines how token-based authorization and resource governance works:

- Each twin must verify token quota before activation
- Quota logic is protocol-verified (no runtime override)
- TCI is publicly referenceable and cryptographically hashed
- Prevents overuse, runaway compute, and multi-tenant interference

Note: Detailed TCI protocol definitions, cryptographic interface bindings, and failover handling are available in Fork Blueprint §4.4.

6. Certification Relevance

All certified forks must declare their token logic and module layout. Token quota enforcement and modular traceability are mandatory for compliance. The use of TCI is required for all edge or distributed deployments.