Executive Summary

synTagma: Structural addressing replaces hash-based identity

SSCCS Foundation

The problem: hash-based addressing is the hidden tax on every system

Every content-addressable system today generates identifiers through a hash function – SHA-256, UUID, hash tables. This cost is so universal that it has become invisible: ~10,000 gates and 64–75 cycles for a single SHA-256 operation, followed by collision resolution and dynamic resizing. UUID generation requires entropy and delivers only probabilistic uniqueness. The industry's response has been faster hash units and larger hash tables – compensating for a structural inefficiency rather than eliminating it.

This tax scales with data. A hash table that performs well at 10,000 entries degrades measurably at 10 million. An index that supports exact lookup provides no structural prefix query. Every new system inherits the same bottleneck.

A structural coordinate space

synTagma replaces hashing with a fixed 16-bit, 3-axis composition space: a contiguous address block whose composition formula embeds three independent structural axes. A combinational decoder of approximately 300 gates extracts the three axis fields from any valid 16-bit value in a single cycle, with zero collisions and zero hash computation. Of 65,536 possible values, exactly 11,172 are structurally valid and the remaining 54,364 are detectable as invalid at the hardware level.

Multi-Coord composition extends the address space without modifying the decoder. Six Coords exceed typical distributed system requirements; nineteen match SHA-256's 2^256 space. Each axis position can represent an application-defined dimension – region, device type, timestamp, shard – making the address itself a structured coordinate.

Measured performance: vs. Rust HashMap

Single lookup: 0.39 ns vs 8.50–227 ns (10–582x)
Nonexistent prefix (10M): 1.65 ns vs 23.05 ms (14Mx)
Compound axis filter: 329 Melem/s vs 2.4 Melem/s (137x)
Bulk operations (11K): 26.4 µs vs 385 µs (14.6x)
500K interleaved ops: 3.64 ms vs 12.2 ms (3.4x)

Lookup latency is flat across all depths in the native path: 0.39 ns at every N, because every Coord resolves to a single array load. The tree fallback scales linearly with N (2.69 ns at N=3, 58.6 ns at N=19). Neither path depends on data volume – 10^4 and 10^77 entries cost the same number of dereferences.

Memory efficiency follows a different curve from hash tables. Dense preallocation (fixed 119 MB for the full coordinate grid) drops to 11.9 B/entry at 10M entries. Tree allocation scales with prefix count, not entry count: each leaf node covers 11,172 slots regardless of occupancy. Hash table memory and allocation calls grow linearly with every entry and exceed both strategies at scale.

Application domains

LLM inference cache. KV caches indexed by token prefixes achieve O(N) direct array access with zero hash computation. Production cache sizes ($10^{4-10}7$ entries) are covered by 2–4 Coords.

Embedded systems. The 11,172-identifier space fits in a single 22 KB no-allocator array. Every coordinate is a direct array index: one load, no hashing, no collisions, no resizing.

Graph and multi-dimensional query. Each node maps to a coordinate, each edge type to a bit array. Adjacency reduces to a bitwise AND over 175 machine words – no index intersection.

General-purpose addressing. Wherever UUIDs, hash keys, or sequence numbers are used, synTagma provides a shorter, faster, deterministic alternative with zero collision probability.

Status

synTagma is an open-source project (Apache 2.0) under the SSCCS Foundation. The Rust reference implementation is available on GitHub with a full benchmark suite and end-to-end verification harness. The coordinate space is exhaustively verifiable – all 65,536 inputs can be validated against the decoder specification in milliseconds on any commodity system, a guarantee that no hash-based system can offer.

Commercial licensing and hardware implementation partnerships are available. For inquiries, demos, or partnership discussions: **synTagma@ssccs.org**