Krystal Primitive Specification

Version 1.0 (KKS-1.0 / KRL-1.0 / KRC-0)

Date: 2025-12-27

Purpose: define a deterministic, offline-verifiable memory primitive: time coordinates + content addressing + append-only registry.

1. Overview

Krystal is a deterministic memory primitive built from three pieces:

• **KKS-1.0**: maps an integer *pulse* into (dayIndex, beat, stepIndex, pulseInStep) using pure integer math, with exact daily closure.

• **KRL-1.0**: shareable locators for moments (stream payloads) and artifacts (content hashes), optionally carrying a compact capsule.

• **KRC-0**: minimal registry format: a JSON file containing a single ordered array of KRL strings.

A conforming implementation can verify coherence from bytes alone: decode locators, derive coordinates from pulse, and detect mismatches.

2. KKS-1.0: Kairos Lattice and Closure

KKS-1.0 maps a non-negative integer pulse into a daily lattice of 36 beats x 44 steps x 11 pulses = 17,424 grid-pulses/day, while closing the day exactly at 17,491.270421 pulses/day.

Constant	Value
MICRO	1,000,000 (µpulses per pulse)
N_DAY_MU	17,491,270,421 (µpulses/day)
BEATS_PER_DAY	36
STEPS_PER_BEAT	44
PULSES_PER_STEP	11
GRID_PULSES_PER_DAY	17,424 (= 36*44*11)

Normative mapping (integer math):

```
P_MU      = pulse * MICRO
dayIndex  = P_MU // N_DAY_MU
r_MU      = P_MU - dayIndex * N_DAY_MU
gridIndex = (r_MU * GRID_PULSES_PER_DAY) // N_DAY_MU

beat        = gridIndex // 484
stepIndex   = (gridIndex % 484) // 11
pulseInStep = gridIndex % 11
```

Test vector: pulse 9,777,777 -> beat 0, stepIndex 14.

3. KRL-1.0: Resource Locators

KRL-1.0 defines how to decode two locator families:

• **Stream locators**: embed a JSON moment as base64url JSON.

- Path form: /stream/p/{b64url_json}

- Fragment form: /stream#t={b64url_json}

• **Content locators**: address an artifact by SHA-256 hex and optionally include metadata in query param p.

- /s/{hex_sha256}?p={b64url_json} (expanded)

- /s/{hex_sha256}?p=c:{b64url_json} (capsule)

If a decoded payload claims pulse + beat/stepIndex, a verifier recomputes coordinates via KKS-1.0 and rejects mismatches.

4. KRC-0: Registry File

A KRC-0 registry is a JSON object containing an ordered list of KRL strings.

```
{
  "urls": ["...","..."]
}
```

Order is significant and SHOULD be treated as append-only history.

5. Canonicalization and Hashing

KCS-1 defines canonical JSON bytes (UTF-8, sorted keys, no whitespace, minimal escaping). KHS-1 defines SHA-256 hashing in lowercase hex.

Artifact hash:

```
artifactHash = sha256(raw_bytes)  # 64 lowercase hex chars
```

Object hash:

```
objectHash = sha256( KCS1(json_object_bytes) )
```

6. Reference implementation

This repository includes a Python reference implementation and CLI:

```
krystal kks 9777777
krystal decode-url <url>
krystal verify-registry <registry.json>
krystal normalize-registry <in.json> <out.json>
```

Conformance is enforced by test vectors in test-vectors/ and pytest tests in reference/python/tests/.