

OpenZL x Axiom — Technical Cheat Sheet (Exhaustive)

Generated: 2025-10-07 22:57:46

Core Objects (Python)

Class	Key Methods/Attrs
Input	content, content_size, elt_width, num_elts, string_lens, type, __init__, __new__, get_int_metadata, set_int_meta
Buffer	content_size, elt_width, num_elts, type, __init__, __new__, as_bytes, as_dltensor, as_ndarray, as_pytensor
Output	content, content_capacity, content_size, elt_width, elts_capacity, mut_content, mut_string_lens, num_elts, string
MutBuffer	content_size, elt_width, num_elts, type, __init__, __new__, as_dltensor, as_ndarray, as_pytensor
Compressor	build_graph, validate, set/get parameter
Compress	compress(), compress_typed(), compress_multi_typed()
Decompress	decompress(), decompress_tbuffer(), decompress_multi_tbuffer()
Selector/FunctionGraph	Selector.select(), SelectorState.parameterize_destination(), GraphState.run_node(), set_destination()

C API — ZL_Compressor

Lifetime Management: ZL_Compressor, ZL_Compressor_create, ZL_Compressor_free
Errors & Warnings: ZL_Compressor_getErrorContextString, ZL_Compressor_getErrorContextString_fromError, ZL_Compressor_getWarnings
Parameterization: ZL_Compressor_setParameter, ZL_Compressor_getParameter
Static Graph Creation: ZL_StaticGraphParameters, ZL_StaticGraphDesc, ZL_Compressor_buildStaticGraph, ZL_Compressor_registerStaticGraph_fromNode1o, ZL_Compressor_registerStaticGraph_fromPipelineNodes1o, ZL_Compressor_registerStaticGraph_fromNode, ZL_Compressor_registerStaticGraph, ZL_NODELIST, ZL_GRAPHLIST
Node Customization: ZL_NodeParameters, ZL_ParameterizedNodeDesc, ZL_Compressor_parameterizeNode, ZL_Compressor_registerParameterizedNode, ZL_Compressor_cloneNode
Graph Customization: ZL_GraphParameters_s, ZL_ParameterizedGraphDesc, ZL_GraphParameters, ZL_Compressor_parameterizeGraph, ZL_Compressor_registerParameterizedGraph
Graph Component Lookup: ZL_Compressor_getNode, ZL_Compressor_getGraph
Graph Finalization: ZL_Compressor_selectStartingGraphID, ZL_Compressor_validate
Reference Compressor: ZL_CCtx_refCompressor, ZL_CCtx_selectStartingGraphID

C API — Compress

Lifetime Management: ZL_CCtx_create, ZL_CCtx_free
CCtx Compression: ZL_compressBound, ZL_CCtx_compress, ZL_CCtx_compressTypedRef, ZL_CCtx_compressMultiTypedRef
Errors & Warnings: ZL_CCtx_getErrorContextString, ZL_CCtx_getErrorContextString_fromError, ZL_CCtx_getWarnings
Typed Inputs: ZL_TypedRef_createSerial, ZL_TypedRef_createStruct, ZL_TypedRef_createNumeric, ZL_TypedRef_createString, ZL_TypedRef_free
Compression Parameters: ZL_CCtx_setParameter, ZL_CCtx_getParameter, ZL_CCtx_resetParameters, ZL_CCtx_setDataArena, ZL_CParam

C API — Decompress

Simple API: ZL_decompress
Querying Compressed Frames: ZL_getCompressedSize, ZL_FrameInfo_create, ZL_FrameInfo_free, ZL_FrameInfo_getFormatVersion, ZL_FrameInfo_getNumOutputs, ZL_FrameInfo_getOutputType,

ZL_FrameInfo_getDecompressedSize
 Helper Functions: ZL_getDecompressedSize, ZL_getNumOutputs, ZL_getOutputType
 Lifetime Management: ZL_DCtx_create, ZL_DCtx_free
 Parameterization: ZL_DCtx_setParameter, ZL_DCtx_getParameter, ZL_DCtx_resetParameters
 Errors & Warnings: ZL_DCtx_getErrorContextString, ZL_DCtx_getErrorContextString_fromError, ZL_DCtx_getWarnings
 Serial Decompression: ZL_DCtx_decompress
 Typed Decompression: ZL_DCtx_decompressTBuffer, ZL_DCtx_decompressMultiTBuffer
 ZL_TypedBuffer: ZL_TypedBuffer_create, ZL_TypedBuffer_createWrapSerial, ZL_TypedBuffer_createWrapStruct, ZL_TypedBuffer_createWrapNumeric, ZL_TypedBuffer_free, ZL_TypedBuffer_type, ZL_TypedBuffer_rPtr, ZL_TypedBuffer_byteSize, ZL_TypedBuffer_numElts, ZL_TypedBuffer_eltWidth, ZL_TypedBuffer_rStringLens

Codecs — At a Glance

Codec	Use / Notes
ACE	Automatic Column Explorer (trainer-driven column grouping & transform search).
Bitpack	Bit-level packing of fixed-width fields.
Bitunpack	Inverse of Bitpack (used in parse flows).
Brute Force	Trainer utility to exhaustively test transform/param combos under a budget.
Compress	Inline zstd (fallback) or nested compression on sub-streams.
Concat	Concatenate sub-streams, often after tokenization/splitting.
Constant	Recognize constant-valued fields; encode once, reference many.
Conversion	Type/endianness conversion when parsing to numeric/struct.
SDDL	Parse bytes to typed fields using Simple Data Description Language.
Dedup	Field-level deduplication / dictionary extraction.
Delta	Delta coding for near-sorted or smooth numeric sequences.
Dispatch	Branch selection by runtime statistics (control points).
Divide By	Scale numeric sequences to tighten ranges.
Entropy (Huffman/FSE)	Final entropy coding; order-0 Huffman or FSE.
Field Lz	LZ matchfinding at field granularity with entropy back-end.
Flatpack	Lay out structs contiguously (AoS↔SoA transforms).
Float Deconstruct	Expose sign/exponent/mantissa to improve coding.
Merge Sorted	Merge multiple sorted lists preserving order metadata.
Parse Int	Fast integer parsing from text inputs.
Prefix	Common-prefix factoring for strings/binaries.
Quantize	Lossy step for bounded numerics (if configured for lossy use-cases).
Range Pack	Base+range packing for bounded numeric values.
Split By Struct	Split arrays of structs into columns (SoA).
Split	Split a stream by token/range/type.
Store	Unit graph: store data verbatim in frame (no transform).
Tokenize	Build dictionary + index vectors for low-cardinality strings.
Transpose	Transpose bytes/bits to cluster significant bits.

Zigzag	Zigzag map for small signed integers.
Zstd	General-purpose zstd; universal fallback.

Selectors & Function Graphs

- Selector.select(state, input) -> GraphID
- SelectorState: get_cparam(), get_local_(int_)param(), parameterize_destination()
- FunctionGraph.graph(state), GraphState.run_node()/run_multi_input_node(), set_destination()

CLI Snippets

```
# Profiles
zli compress --profile serial in.txt -o out.zl
zli compress --profile le-i32 ints.bin -o out.zl

# Train & use
zli train --profile csv samples/ -o trained.zsc
zli compress --compressor trained.zsc sample.csv -o out.zl
```

AxiomZL Wrappers (Proposed)

Function	Purpose
axiom_zl.load_profile(name) -> Compressor	Load stock OpenZL profile (serial, le-i32, csv) with Axiom defaults.
axiom_zl.describe_sddl(sddl_str) -> Compressor	Build a compressor from SDDL text; auto-wire I/O types.
axiom_zl.train(compressor, samples, budget, targets) -> Compressor	Offline training orchestration with budget & speed/ratio targets.
axiom_zl.run(in_bytes, compressor) -> bytes	One-shot compress (typed refs inferred if possible).
axiom_zl.decompress(frame_bytes) -> dict[str, bytes ndarray]	Universal decoder wrapper; returns typed outputs when available.
axiom_zl.graph().split().tokenize().entropy()...	Fluent builder mirroring FunctionGraph & Selector APIs.
axiom_zl.selector(metric=...) -> branch()	Selector with Axiom heuristics: pick branch via cheap stats.
axiom_zl.params(level=?, format_version=?, data_arena=?)	Unified parameter DSL mapped to ZL_CParam / LocalParams.
axiom_zl.inspect(frame) -> info	Expose ZL_FrameInfo via friendly object.
axiom_zl.trace(frame) -> plan	Extract and pretty-print resolved graph for observability.

Heuristics (Quick Picks)

- numeric monotone: delta → transpose → entropy
- low-card strings: tokenize → (dict zstd) & (idx entropy)
- bounded floats: float-deconstruct → range-pack → entropy
- unknown/texty: fall back to zstd