RAMTERA TECHNICAL WHITE PAPER SERIES

WP-10

OASIS Conformance, Record by Record

What a complete SEMI P39 reader and writer actually has to implement

A format claim is only useful if it says which records are handled and which are not. Here are both lists.

34	12	18	17
RECORD TYPES READ	REPETITION TYPES	TRAPEZOID FORMS	RECORD TYPES WRITTEN

Document	WP-10 — OASIS Conformance, Record by Record
Author	Sameer Pawanekar, Founder & Chief Executive Officer, Ramtera Inc.
Revision	v1.0
Date	August 2026
Classification	Public

ABSTRACT

OASIS replaced GDSII as the layout interchange format because GDSII files had become impractically large. It achieves that through modal variables, repetition encoding, compact geometry primitives and block compression — each of which is a place an implementation can be incomplete without being obviously wrong. This paper enumerates what a conforming reader and writer must handle, reports our implementation's coverage record type by record type, and states explicitly which records are read but not written and which are consumed but not preserved. The asymmetries are deliberate and are named rather than omitted.

01 Why OASIS is harder than it looks

GDSII is a simple format: a stream of self-describing records with absolute coordinates. A minimal reader is a weekend's work, and a minimal reader is very nearly a complete one.

OASIS is not like this. Its compactness comes from encodings that carry state, and a reader that handles each record type in isolation without maintaining that state will parse a file successfully and produce wrong geometry.

Mechanism	What it does	Where implementations go wrong
Modal variables	Fields omitted from a record inherit the last value used	Reset scope. Getting it wrong shifts geometry by whatever the stale value was.
Repetition encoding	One record describes many placements or shapes	Twelve distinct repetition types, several with non-obvious count encodings
Variable-length integers	Coordinates and counts in a compact byte encoding	Signed and unsigned forms; delta encodings with directional variants
Compact geometry	Trapezoids and rectangles encoded by type rather than by vertex list	Eighteen trapezoid forms, each with its own vertex derivation
Block compression	Cell contents compressed inline within the stream	Requires a nested stream abstraction; a flat reader cannot handle it
Name tables	Strings referenced by number, with implicit and explicit numbering	Both forms are legal and a reader must accept either

Table 1 — OASIS mechanisms and their failure modes.

MODAL RESET SCOPE IS WHERE INTEROPERABILITY ACTUALLY BREAKS

Modal variables deserve particular attention because their reset scope is a convention rather than a strict requirement, and tools differ. Our implementation performs a full modal reset at every cell boundary, matching the prevailing convention among established tools. A reader that assumes a different scope will read files written by those tools incorrectly, and vice versa.

02 Reader coverage

The reader handles every record type defined by the format. The table below lists them by category, with the handling stated where it is anything other than full interpretation.

Category	Records	Handling
Structural	PAD, START, END	Full, including offset table and end-record checksum
Name tables	CELLNAME, TEXTSTRING, PROPNAME, PROPSTRING	Full, both implicit and explicit reference numbering
Layer names	LAYERNAME for data and text	Full, including interval and singleton layer/datatype specifications
Cell structure	CELL by reference and by name	Full
Coordinate mode	XYABSOLUTE, XYRELATIVE	Full
Placement	PLACEMENT by reference and by name	Full, including magnification, angle and reflection

Category	Records	Handling
Geometry	RECTANGLE, POLYGON, PATH, TEXT, CIRCLE, TRAPEZOID A/B/C	Full; 18 compact trapezoid forms derived to explicit vertices
Properties	PROPERTY, LAST_PROPERTY	Parsed; net-name properties extracted and retained
Extension	XNAME, XELEMENT, XGEOMETRY	**Consumed and skipped.** Correctly advances the stream; content not retained
Compression	CBLOCK	Full, zlib compression type, with nested stream handling
Repetition	All 12 types	Fully expanded to individual instances

Table 2 — Reader coverage. All 34 defined record types are handled.

Geometric interpretation decisions

Two reader behaviours are interpretation choices rather than conformance requirements, and both affect what downstream verification sees. They are stated here because a reader that makes different choices will produce different verification results from the same file.

- **Circles are converted to octagons.** A verification engine operating on polygon sets has no native circle. The conversion follows the eight-sided convention used by established verification tools, so that results correlate. A circle is therefore not preserved as a circle through a round trip.
- **Manhattan and non-Manhattan polygons take different paths.** Axis-aligned polygons are decomposed for efficient scanline processing; polygons containing angled edges are retained as general polygons without decomposition, so that angled geometry — seal rings, crack stops — survives ingestion intact.
- **Paths are expanded to polygons** and also retained in path form, so that rules referring to either representation can be executed.
- **Winding order is normalised** before geometry enters the polygon engine.

03 Writer coverage, and the asymmetries

The writer emits seventeen record types. It is deliberately narrower than the reader, and the gaps are listed rather than left for a user to discover.

Records emitted	Notes
PAD, START, END	Full 13-entry offset table; CRC32 checksum over the correct byte range
CELLNAME, TEXTSTRING, PROPNAME (explicit form)	Explicit reference numbering only — see asymmetries below
CELL by reference, XYABSOLUTE	Absolute coordinates only; modal variables fully reset per cell
RECTANGLE, POLYGON, PATH, CIRCLE, TRAPEZOID (compact form)	Geometry set
TEXT, PROPERTY	Text and property emission
PLACEMENT by reference	Includes repetition detection and encoding
CBLOCK	Cells above a size threshold are emitted compressed

Table 3 — Writer coverage.

What the writer does not emit

Record or form	Read?	Written?	Consequence
Implicit reference numbering in name tables	Yes	No	Output uses explicit numbering only. Both are legal; explicit is unambiguous.
XYRELATIVE coordinate mode	Yes	No	Output is absolute-coordinate throughout. Larger, unambiguous.
PLACEMENT by name	Yes	No	Output uses reference form only.
TRAPEZOID A and B forms	Yes	No	Emitted in the compact trapezoid form instead.
PROPSTRING table	Yes	No	Property strings emitted inline.
XNAME, XELEMENT, XGEOMETRY	Consumed	No	**Extension records are not preserved through a round trip.**
CIRCLE as a circle	Yes	As geometry	Circles are octagons after ingestion; the original circle is not recovered.

Table 4 — Reader/writer asymmetries, stated explicitly.

WHY THE WRITER IS NARROWER THAN THE READER

Every asymmetry above trades file size for unambiguity. Absolute coordinates, explicit reference numbering and a single trapezoid form all produce larger files than the alternatives and remove a class of encoding ambiguity that is otherwise a source of interoperability defects. The two genuine losses — extension records and circle identity — are named as losses rather than presented as choices.

04 How conformance should be tested

A record-type checklist is necessary and not sufficient. The tests that establish conformance in practice:

1. **Geometric round trip.** Read, write, re-read, and XOR against the source per layer. An empty difference on every layer is the acceptance criterion. This catches encoding errors that a structural comparison misses entirely.
2. **Cross-tool round trip.** Read a file written by an independent implementation, and have an independent implementation read yours. A round trip through a single implementation cancels symmetric errors.
3. **Repetition stress.** All twelve repetition types, including the irregular and diagonal forms, with counts near the encoding boundaries. Repetition count encodings differ between types and the off-by-one cases are where implementations diverge.
4. **Compression boundary.** Cells immediately above and below the compression threshold, so that both the compressed and uncompressed paths are exercised in the same file.
5. **Modal reset.** Files in which a cell omits a field that the previous cell set. This is the single most productive test for scope errors and is rarely present in synthetic test files.

THE SINGLE MOST VALUABLE TEST

The geometric round trip is the test that matters most, because it is the only one that validates the encoding and the interpretation together. A reader can handle every record type on the checklist and still place geometry incorrectly.

05 Conclusion

OASIS conformance is not a binary property and is poorly described by a claim of support. An implementation handles a specific set of record types, makes specific interpretation decisions about geometry that has no direct representation in the consuming engine, and emits a specific and usually narrower subset than it reads.

The useful form of the claim is the one made here: the reader handles all thirty-four defined record types, all twelve repetition types and all eighteen compact trapezoid forms; the writer emits seventeen record types in unambiguous forms; extension records and circle identity are not preserved through a round trip. A user can act on that. They cannot act on a checkmark.

About Ramtera

Ramtera Inc. develops **LVR**, a physical verification engine that executes foundry SVRF rule decks natively, covering design rule checking, layout versus schematic, electrical rule checking, layout XOR, density and fill, and parasitic extraction. LVR is written in modern C++ with a Qt6 interface, and has been validated against publicly available process design kits including SKY130 and IHP SG13G2.

Ramtera also develops **PRE**, a place-and-route engine sharing LVR's geometry kernel, and maintains a portfolio of granted patents, registered copyrights and trademarks across its verification and implementation technologies.

Ramtera Inc. is incorporated in the United States, with research and development conducted by Ramtera Design Automation India Private Limited in Bangalore, India.

Contact	
Web	https://ramtera.com
General enquiries	info@ramtera.com
Technical support	support@ramtera.com
Series	Ramtera Technical White Paper Series
This document	WP-10, revision v1.0, August 2026