RAMTERA

PRODUCT DOCUMENTATION · PHYSICAL VERIFICATION (LVR)

RAMTERA-LVR-ARCH

Platform Architecture Overview

One design model, five engines, deterministic parallel execution

August 2026 · Ramtera PRE engineering

LVR Platform Architecture Overview

This document describes LVR's architecture at the level a CAD team needs for deployment and flow-integration decisions: the compilation pipeline from rule deck to execution, the engines that share one design model, and the concurrency and reproducibility properties the platform guarantees.

1	native	deterministic	tiled
shared design model, five engines	deck execution — no intermediate translation	results at any thread count	spatial parallel execution

From rule deck to execution

LVR's front end is a compiled parser for industry-standard rule deck syntax — a grammar measured in the millions of lines, engineered for fast builds and strict conformance (see white paper WP-01). Decks execute natively: the parsed rule graph drives the geometry engines directly, with derived-layer dependencies resolved once and reused across checks. Preprocessor conditionals, layer maps, variable definitions, and check-selection groups behave as production decks expect.

One design model, five engines

Layout is read once — hierarchical GDSII or OASIS — into a single in-memory model with exact database-unit semantics. Connectivity is extracted once and shared: LVS, ERC, and PEX all consume the same connectivity model, which is why a combined DRC/LVS/PEX run costs far less than three tools' runs. Geometry operations dispatch across multiple kernels suited to rectilinear, 45-degree, and arbitrary-angle content.

Concurrency and reproducibility

Property	Statement
Spatial tiling	large designs execute as spatial tiles processed by a worker pool; scaling on reference workloads is near-linear through the tested core counts (white paper WP-12)
Determinism	results are independent of thread count and of run-to-run scheduling — checked routinely by comparing result databases across configurations
Memory dial	peak memory scales with thread count × tile working set; reducing threads trades runtime for footprint on small machines, predictably
Distribution	farm execution via standard batch schedulers; the reference full-chip cloud run used 128 instances (1,024 vCPU)

Results, waivers, and debug

Violations land in a results database browsable by rule, cell, and location, with mismatch-diagnosis panes for LVS. Waivers persist across layout edits and ECOs — a waived violation stays waived when its cell moves — and are stored separately from results so waiver review is auditable. Violations can be pushed as native markers into major custom-IC layout environments and read back for waiver assignment.

SCOPE OF THIS DOCUMENT — This overview intentionally describes properties, not internal mechanisms. Several of the mechanisms behind the tiling, determinism, waiver-identity, and kernel-dispatch properties are the subject of patent filings.

About Ramtera

Ramtera Inc. (Delaware, USA), with R&D at Ramtera Design Automation India Pvt Ltd (Bangalore), builds electronic design automation software for physical verification and physical implementation. LVR is Ramtera's physical verification engine (DRC, LVS, ERC, XOR, parasitic extraction); PRE is Ramtera's physical implementation environment (floorplanning, placement, clock tree synthesis, routing, timing analysis and optimization), built on industry-standard command conventions so existing flow scripts and operator skills transfer directly. The company holds granted patents and registered copyrights covering its engines.

Contact

Web	www.ramtera.com
Inquiries	sameer@ramtera.com
Engineering	Bangalore, Karnataka, India

This document reports measurements from Ramtera's engineering environment (Ubuntu 24.04, GCC 13.3, Tcl 8.6) unless stated otherwise. Product behavior and figures are subject to change; no compatibility, certification, or endorsement by any third party is stated or implied.