cadence®

Incisive Formal Verifier

Assertion-based verification and debug of RTL block designs

Incisive® Formal Verifier brings the power of formal analysis to your desktop, resulting in significant productivity gains and improved design quality. As a key component of the Incisive platform's complete assertion-based verification solution, Incisive Formal Verifier enables you to begin verification months earlier, reducing re-spins and speeding up time-to-market.

Incisive Verification Platform

The functional verification of nanometer-scale ICs requires speed and efficiency. Yet today's fragmented methodologies make it impossible to optimize either. Each verification stage has its own methodology, tools, models, and user interface. Engineers must re-create almost everything at every stage. The Cadence Incisive verification platform is the world's first functional verification platform that combines formal analysis, simulation, acceleration and emulation, and supports a unified methodology to deliver the fastest, most efficient verification in the industry.

Incisive Formal Verifier

Incisive Formal Verifier uses the industry's most advanced formal analysis technology to offer design teams superior performance, capacity, and ease-of-adoption. With its robust, production-proven technology, Incisive Formal Verifier enhances both productivity and product quality.

Part of the Incisive platform's complete assertion-based verification solution, Incisive Formal Verifier supports the same set of assertions as Incisive simulation, coverage, acceleration, and emulation technologies. An effective methodology, developed through extensive collaboration with customers, delivers the most efficient way to adopt assertion-based verification and formal analysis, maximizing your return on investment.

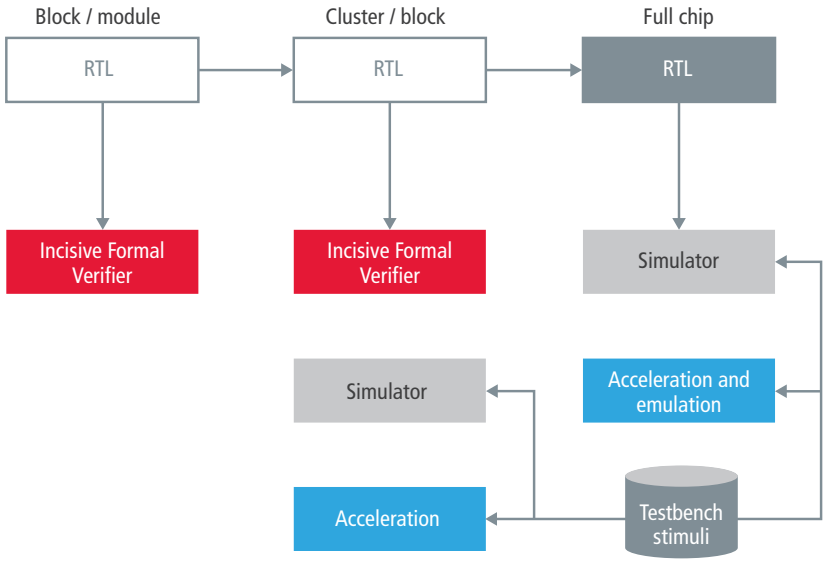

Figure 1: Incisive Formal Verifier brings formal analysis to the designer's desktop

Since Incisive Formal Verifier does not require a testbench, you can begin verification months earlier when designing the RTL blocks. Formal methods also pin-point the source of each exposed bug, reducing block debug and integration time. Due to its exhaustive analysis capabilities, Incisive Formal Verifier can detect hard-to-find corner-case functional bugs, many

of which would be very expensive or even impossible to find using traditional techniques. The net result is a significant gain in productivity, which minimizes the risk of re-spins, improves design quality, and accelerates time-to-market.

Benefits

- Speeds time-to-market with lower risk and higher predictability
- Increases productivity by enabling verification to start months earlier, before testbench development and simulation
- Improves quality and reduces risk of re-spins by exposing corner-case functional bugs that are difficult or impossible to find using conventional methods
- Reduces block design effort and debug time, and shortens integration time
- Provides design teams with an advanced debug environment with simulation synergies for ease-of-adoption

Features

Leading-edge formal engines with smart automation

A set of complementary, state-of-art formal engines from production-proven technologies and world-renowned researchers combine to deliver the industry's highest performance and capacity.

To ensure ease-of-use and maximize performance, a strategy engine automates the engine selection process by selecting optimal engines for each run. To optimize formal analysis performance even further, built-in distributed processing allows you to leverage the broad engine assortment by running them in parallel for multi-fold performance gains.

Broad design language support

Incisive Formal Verifier provides extensive design language support, including Verilog, SystemVerilog, VHDL, and mixed-language, to leverage formal analysis across all design teams and groups within your company. Using mature and robust front-end parsers proven on thousands of designs provides you with improved reliability when deploying Incisive Formal Verifier into your flow.

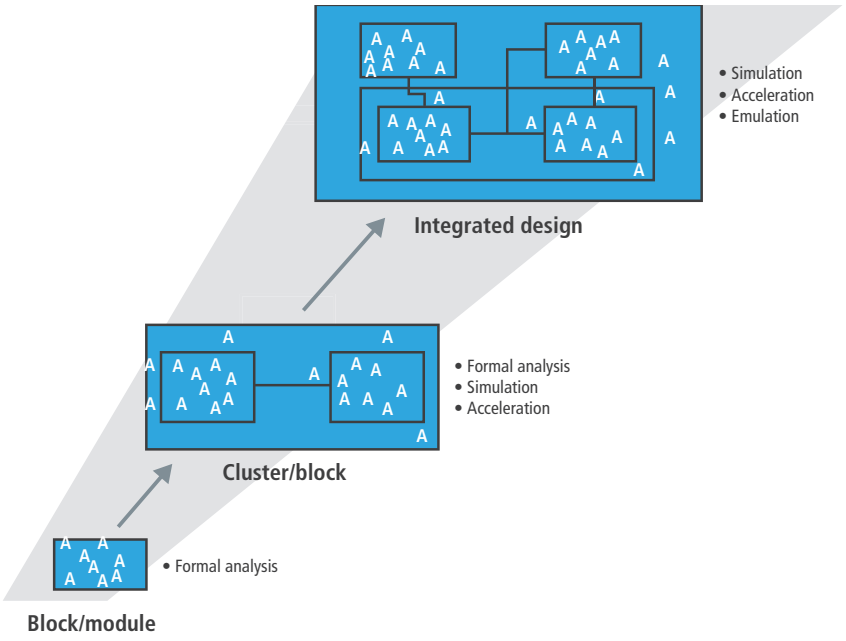

Figure 2: Formal analysis at the entry point of the Incisive platform's complete assertion-based verification flow

Broad assertion support and interoperability

Incisive Formal Verifier supports the same set of assertions as Incisive simulation, acceleration, and emulation support. This includes assertions written in Property Specification Language (PSL), SystemVerilog Assertions (SVA), Open Verification Library (OVL), and the open source Incisive Assertion Library (IAL).

Automatic assertion extraction

Incisive Formal Verifier automates assertion creation for common design structures, with the extraction of assertions for FSM states and arcs, branching deadcode, and synthesis pragma assumptions. Such automation of the simpler assertions allows you to focus on design-specific assertion creation, maximizing your productivity while at the same time ensuring that all functional problems related to common design structures are exposed automatically.

Simulation synergy

Incisive Formal Verifier integrates seamlessly with Incisive Unified Simulator and works great with third-party simulators as well.

The Incisive platform environment uses common parsers, assertions, linting, analysis, coverage, and debug.

Moreover, Incisive Formal Verifier has built-in initialization capabilities for block-level verification. It also leverages the industry-standard VCD interface, supported by all simulators, to use simulation traces when deep or complex initialization is required.

Comprehensive, easy-to-use debug and analysis environment

An integrated GUI environment and Tcl interface provides you with an easy-to-adopt debug and analysis environment — the same as that in the Incisive Unified Simulator and similar to other simulators. The complete environment includes built-in linting, waveform viewing with source code linking, source code value annotation and tracing, structural analysis, vacuity and sanity checks, coverage reporting, and overall verification management.

When Incisive Formal Verifier detects an assertion violation, it generates a simulated counter-example waveform for easy debugging. You can also generate a simple testbench for use in any simulator to validate the exposed functional bug or for regression runs. Together, these capabilities allow design teams to instantly deploy and use Incisive Formal Verifier in their production flows.

Complete formal coverage

Users of formal analysis software frequently need to find out if they have written sufficient assertions to verify the design blocks, to understand how much has been formally verified, and to determine the proof radius. Incisive Formal Verifier supports all these features to ensure efficient verification.

Incisive Formal Verifier includes another set of capabilities, including reachability analysis to verify simulation coverage holes. Synergy exists for functional coverage using assertions and for code coverage constructs such as FSM states, FSM arcs, and branching deadcode. Incisive Formal Verifier exposes coverage-associated functional problems that cannot be found using simulation, acceleration, or emulation, and it gives you a complete coverage picture.

Comprehensive Methodology

Incisive Formal Verifier is an integral part of the comprehensive Incisive assertion-based verification (ABV) flow. To optimize verification efficiency, different technologies need to be used where their strengths are maximized. The Incisive ABV flow with formal analysis methodology has enabled customers to easily deploy assertions to their design teams and reduce overall verification time while enhancing design quality.

Formal analysis is deployed at the beginning of the verification flow as the designers write RTL and assertions — months before meaningful testbench simulation begins. With this approach, functional bugs are detected much earlier in the design cycle and are easier to fix at a significantly lower cost. In addition, corner-case bugs, typically missed using traditional flows, are discovered early by formal analysis, which reduces the risk of re-spins.

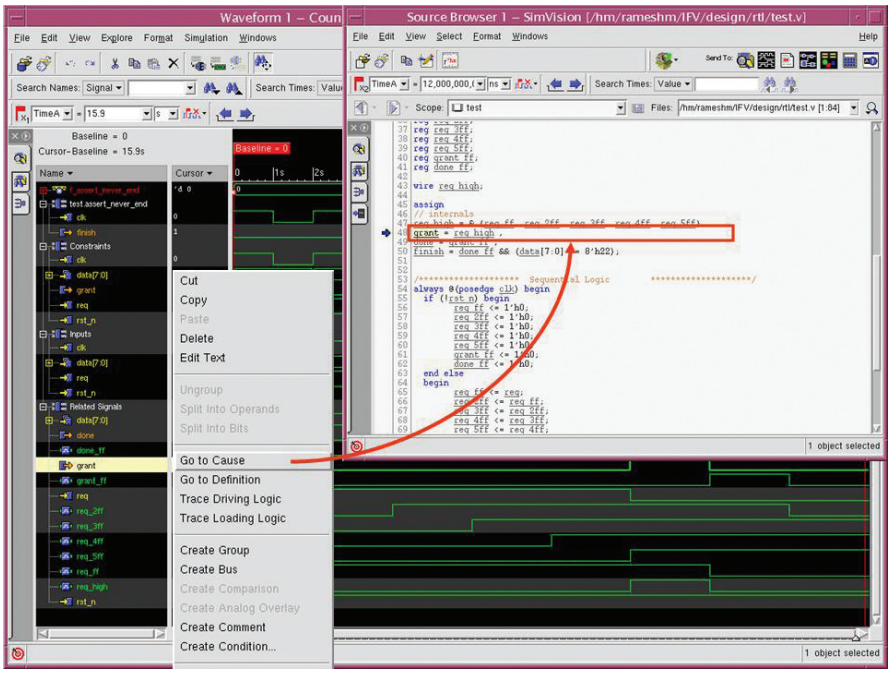

Figure 3: Incisive Formal Verifier provides advanced debug and diagnostics features common to simulation for ease-of-use and ease-of-adoption of formal analysis

The same assertions travel with the blocks and are verified by simulation, acceleration, and emulation later in the design cycle. Since the strengths of each of the technologies comprising the complete ABV flow are used, this methodology is powerful when deployed and easy to incorporate into existing verification flows to improve time-to-market.

Specifications

Design language support

- Verilog (IEEE 1364-1995, IEEE 1364-2001)
- SystemVerilog (IEEE 1800)
- VHDL (IEEE 1076-1987, IEEE 1076-1993)
- Mixed-language environments

Assertion language support

- Property Specification Language (PSL)
- SystemVerilog Assertions (SVA)

Assertion library support

- Open Verification Library (OVL)
- Incisive Assertion Library (IAL)

HDL analysis

- 500+ checks to lint and analyze
  - Synthesizability
  - Race conditions
  - Code reusability
  - FSM coding
  - Verilog, VHDL, and mixed-language support
  - Design subset

Result analysis

- Debug and GUI
  - Assertion manager
  - Waveform viewer
  - Source code browser
  - Source code value annotation
  - Driver and receiver tracing
  - Vacuity and sanity checks
  - Assertion triggering
- Reporting
  - Status and proof radius reporting
  - Assertion coverage reporting
  - Formal coverage reporting
  - Cone-of-influence analysis

Interfaces

- Tcl command interface
- VCD and SST2 interfaces

Platforms

- Sun Solaris
- HP-UX
- Linux

Incisive Product Line-up

Incisive Unified Simulator

- Unified simulator with heterogeneous single-kernel architecture
- Native Verilog, SystemVerilog, VHDL, and SystemC® support
- Native SystemC Verification Library
- PSL, SVA, and OVL support
- Fast, unified test generation
- Acceleration policy checks and HDL analysis
- Unified simulation and debug environment
- Unified kernel supports analog/mixed-signal and algorithm design

Incisive Palladium® family of accelerators/emulators

- Simulation acceleration and in-circuit emulation in one system
- Provides up to 100x-10,000x RTL performance
- Assertion-based acceleration
- Run-time performance with up to 750KHz speed
- Compiles up to 30M gates per hour on a single workstation
- Expandable to 256M gates
- Allows up to 32 simultaneous users
- Leader in microprocessor and IP support

Related Products

Encounter® Conformal® Equivalence Checker

- Equivalence checking capability
- Clock domain crossing checks

Encounter Conformal Constraint Designer

- Functional validation and generation of design constraints

Cadence Services and Support

- Cadence application engineers can answer your technical questions by telephone, email, or Internet—they can also provide technical assistance and custom training
- Cadence certified instructors teach more than 70 courses and bring their real-world experience into the classroom
- More than 25 Internet Learning Series (iLS) online courses allow you the flexibility of training at your own computer via the Internet
- Cadence Online Support gives you 24x7 online access to a knowledgebase of the latest solutions, technical documentation, software downloads, and more

cadence®

Cadence is transforming the global electronics industry through a vision called EDA360. With an application-driven approach to design, our software, hardware, IP, and services help customers realize silicon, SoCs, and complete systems efficiently and profitably. www.cadence.com